A QUEUING MODEL SIMULATION WITH DYNAMIC GRAPHICAL DISPLAY

Cyrus Michael Riddell

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

A QUEUING MODEL SIMULATION

WITH DYNAMIC GRAPHICAL DISPLAY

by

Cyrus Michael Riddell

Thesis Advisor: R. W. Butterworth

March 1973

Approved for public release; distribution unlimited.

A Queuing Model Simulation

with Dynamic Graphical Display

by

Cyrus Michael Riddell
Lieutenant, United States Navy
B.A., University of Washington, 1965

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
March 1973

ABSTRACT

The purpose of this thesis is to provide the student of queuing systems with a vehicle through which a better understanding of the interrelationships between stochastic processes and queuing systems can be achieved.

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF PICTURES

ACKNOWLEDGEMENTS

The Hybrid Computer at the Naval Postgraduate School is a self-operated system; because many of the complexities of the system were unfamiliar to the programmer, the technical assistance of Mr. R. Limes, Mr. W. Thomas, and Mr. A. Wong was invaluable.

I. INTRODUCTION

A queuing system is comprised of action facilities (servers) and waiting facilities (queues or lines). The relationship between the components can be of many forms, each with their own characteristics. Consider a tool crib in a machine shop where the machinists are obliged to check out their tools; or consider the telephone system, in which callers request communication capacity. Both situations involve requests for action from limited resources, in these cases, servers or telephone circuits. In order to allocate resources properly, it is necessary to understand the interaction between requests for action and the actions themselves. Individually these can be modeled as stochastic processes, but in order to predict flow through the queuing system, it is necessary to understand the interrelation between the patterns of event occurrences. When demands occur at regular intervals and actions are of fixed duration, it is relatively easy to predict flow; if there are more requests per unit of time than actions over the same time span, the queue will grow without bound unless the event relationship is changed. When variation is added to the timing of events, the flow is not as obvious. For example, if the average number of requests for action equals the average number of actions, per unit of time, but both occur with variability, then it would be useful to be able to predict such quantities as expected delay and action

facility idle time. Queuing models are used for this purpose.

The purpose of this thesis is to provide the student of queuing systems with a vehicle through which a better understanding of the interrelationships between stochastic processes and queuing systems can be achieved.

This is done through a combination of a computer simulation and a computer-driven graphical display.

II. QUEUING MODEL SIMULATION

This computer program simulates queuing systems in two stages. First, a next-event type computer simulation creates a time history of the queuing system. Second, a dynamic graphical display depicts the occurrence of the queuing process in real time.

Five basic, academically interesting, queuing models, each with parameter options, can be simulated. Queuing models are generally described in terms of the distribution of request occurrences (arrival distribution), the distribution of action occurrences (service distribution), and the number of action facilities (servers). Specifically, the times between request occurrences are taken to be independent and identically distributed realizations from the arrival distribution. The duration of each action occurrence is also taken to be an independent and identically distributed realization from the service distribution. Variations on these assumptions occur for some models and are described in Section A.

The abbreviation G/G/C describes a queuing system in which the arrival and service distributions are general (unspecified) and there are C servers.

The five options in this simulation are:

(1) G/G/C

(2) G/G/C with losses

(3) G/G/C with feedback

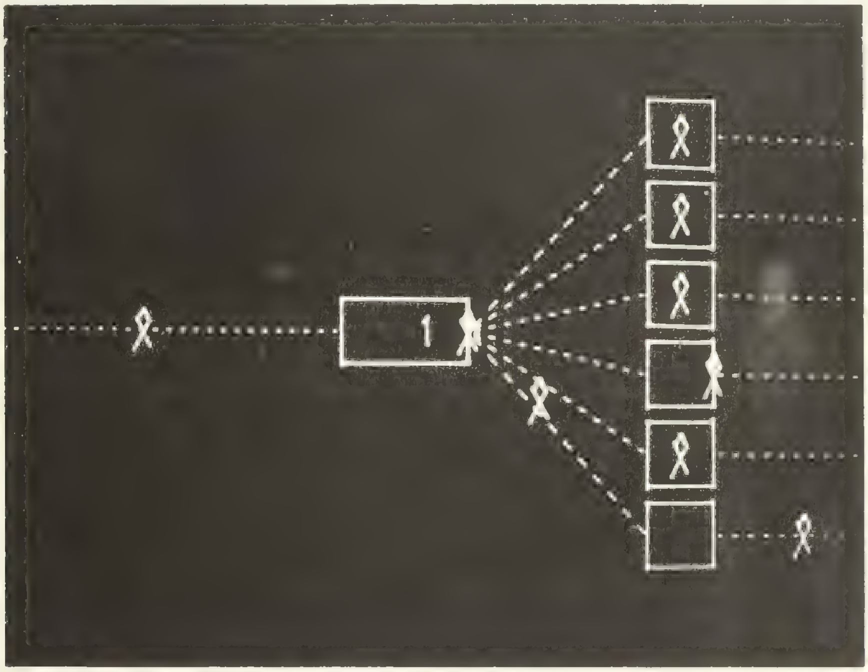
(4) G/G/C with finite source

(5) Two simultaneous G/G/C queues for comparison.

The specific choices of arrival distribution and service distribution which can be simulated are fully described in Section B.

A. THE QUEUING MODELS

1. G/G/C

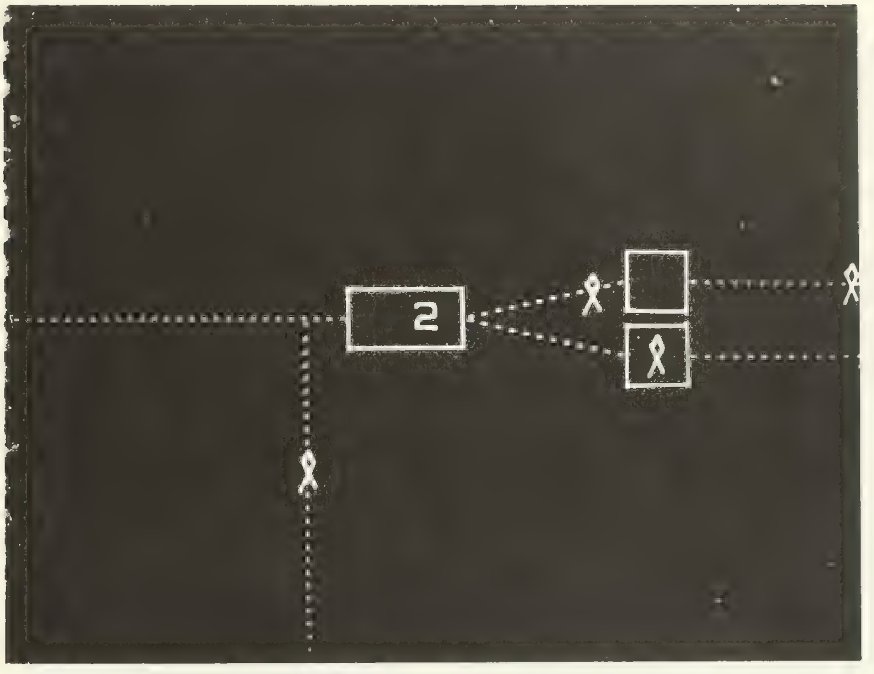


Picture 1

This is the simplest of the queuing models considered here. Each arrival waits in the queue until a server is available, enters service for a calculated period of time, then departs the system.

This model may be simulated with from one to six servers.

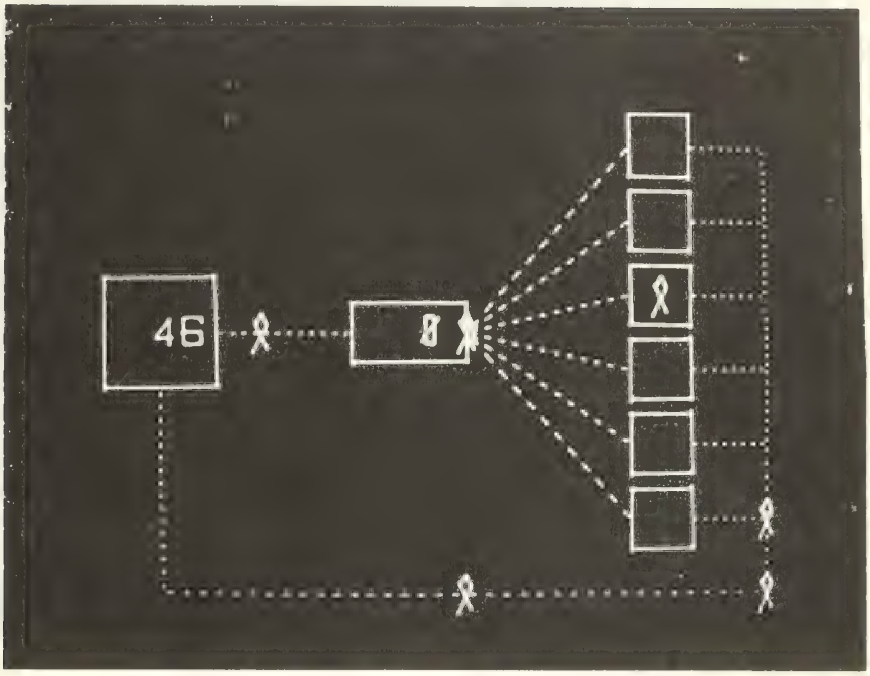
2. G/G/C with Losses



Picture 2

Consider making a long distance telephone call on Mother's Day. If all the trunk lines are busy, your request for service will be rejected. The G/G/C model with losses reflects this by rejecting arrivals who find the queue full. The queue limit used is specified as an input parameter. Note that rejected arrivals are lost and are not assumed to return later. The fraction of all arrivals who are served (not rejected) is recorded for subsequent display.

3. <u>G/G/C with Finite Source</u>

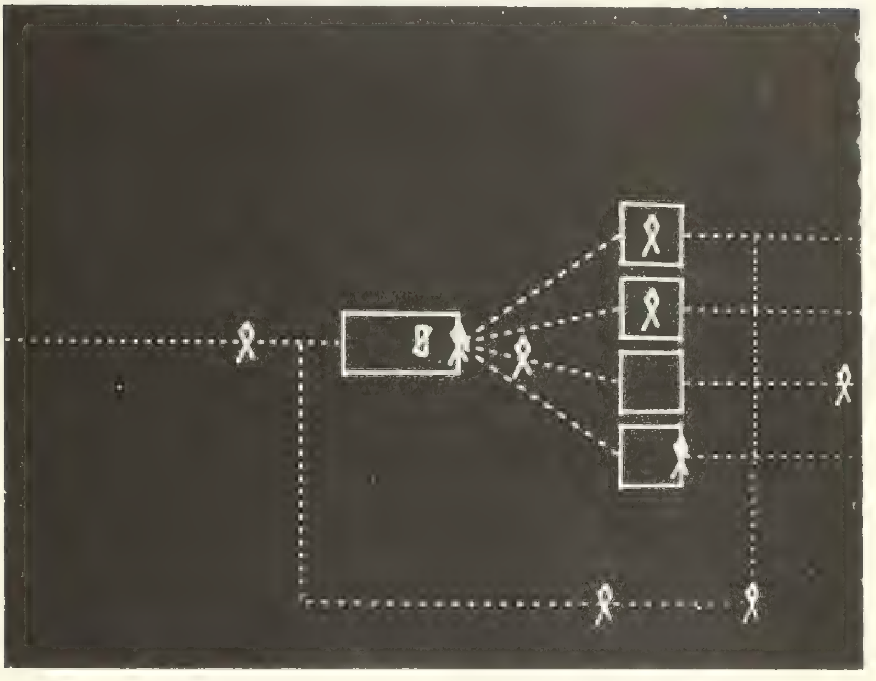


Picture 3

Consider a closed-circuit educational television system at a school. The finite population consists of all the televisions in the system. When one fails, it enters the queue to be serviced. After repair, it re-enters the population pending the next failure.

This situation is modeled by assuming that each action source, upon entering the idle population, is assigned a waiting time from the arrival distribution. When this time expires, the source requests service, joining the queue if necessary. As each action source terminates its service, it returns to the idle population. The user specifies the number of servers (one to six) and the size of the population.

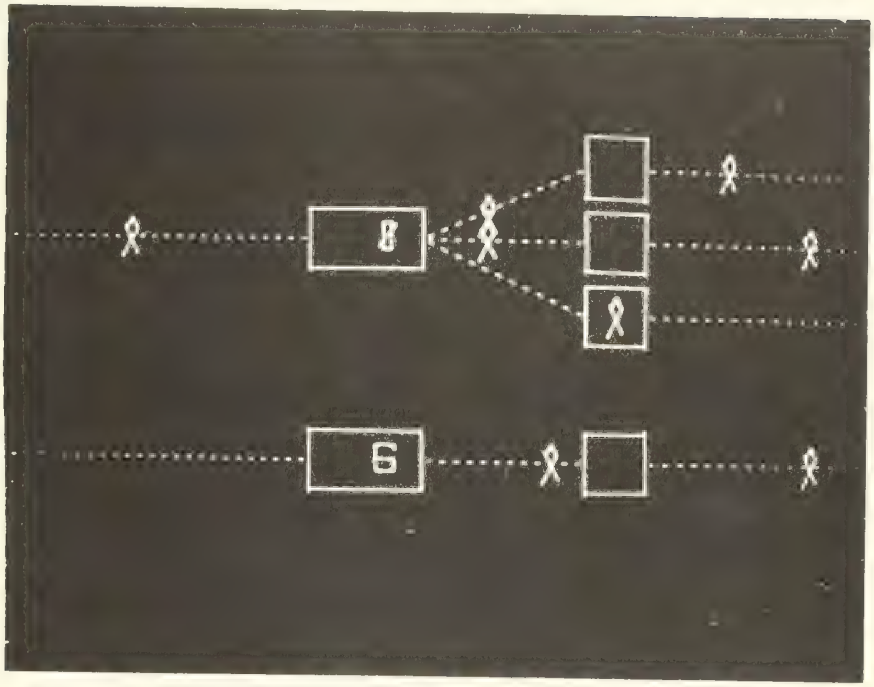
4. G/G/C with Feedback to the Queue



Picture 4

This is an extension of the G/G/C model, with from one to six servers, allowing customers leaving service to depart the system or to return to the queue. At each service completion, the customer leaves the system with probability P or returns to the queue with probability 1-P. The user specifies P as an input option.

5. <u>Two Simultaneous G/G/C Queues</u>



Picture 5

This option allows comparison of two standard G/G/C queues, each with up to three servers. All parameters of each system are specified separately, and, if desired, identical arrival streams may be utilized. This allows comparisons between different arrival and/or service distributions and associated parameters, and between a different number of servers. Also, the effect of the various queuing disciplines can be compared.

B. ARRIVAL/SERVICE DISTRIBUTIONS AND QUEUE DISCIPLINES

As well as options already described, the following arrival and service distributions are available to the user of the program. Any of the following probability

distributions can be used as an arrival or service distribution.

1. <u>Distributions</u>

a. K - ERLANG (exponential when K=1)

$$F(t) = 1 - \sum_{J=0}^{K-1} \frac{(\beta t)^J}{J!} e^{-\beta t}$$

$$\text{RATE} \quad \lambda = \frac{\beta}{K}$$

$$CV = \frac{1}{\sqrt{K}}$$

Input parameters are:

(1) K

(2) Rate - in terms of mean number of events per minute.

b. Hyperexponential

$$F(t) = p(1-e^{-\lambda_1 t}) + q(1-e^{-\lambda_2 t})$$

where:

(1) $p + q = 1$

(2) $CV = \frac{1}{\sqrt{p}}$

(3) λ = Rate is specified and then λ_1, λ_2 chosen to satisfy--

$$\lambda_2 = (2 + \sqrt{2})\lambda$$

$$\lambda_1 = \left(\frac{p(2 + \sqrt{2})}{2 + \sqrt{2} - q}\right)\lambda$$

Input parameters are:

(1) Rate, specified in terms of mean number of events per minute.

(2) Coefficient of variation (must be greater than 1.0)

c. Degenerate (Fixed Rate)

$$F(t) = \begin{cases} 0 & t < 1/\lambda \\ 1 & t \geq 1/\lambda \end{cases}$$

Input parameter:

(1) Rate λ specified in terms of mean number of events per minute.

2. Queuing Disciplines

For comparison purposes, there are four queuing disciplines to choose from. These determine the priority by which members of the queue will enter service.

a. FIFO - the first into queue is the first out. Over all disciplines which are independent of service times, this one minimizes the variance of delay.

b. LIFO - the last into the queue is the first out. Some systems, by their nature, function in this manner.

c. SSTF - the queue member with shortest service time is served first; this minimizes the mean delay over

all other disciplines, but the variance of delay is greater than that of FIFO.

d. RANDOM DRAW - applies to a system in which insufficient information is known to implement any other discipline.

3. Length of the Simulation

The period of simulation is specified in minutes by the user. All queues are empty and servers idle at the start of the simulation.

III. THE METHOD

A. THE COMPUTER SIMULATION

The computer simulation is a modified next event type. There are two types of events, arrivals to the queue and a completion of service. With the first there will be an associated entry to service if there is an idle server, and with the last an associated entry to service if the queue is not empty.

The simulation is begun with the generation of the first arrival time and its associated service time. The times are generated via the transformation of random numbers to stochastic variates of the type specified. At the occurrence of an arrival, the next arrival time and associated service time are generated. When a server is available, a waiting member of the queue enters service with completion time equal to the sum of his assigned service time and the time he departs the queue. Thus with the clock starting at zero, a time history of events is created which tells what happened and when.

At the same time, the appropriate values are retained from which statistics of the simulation can later be calculated.

B. THE GRAPHICAL DISPLAY

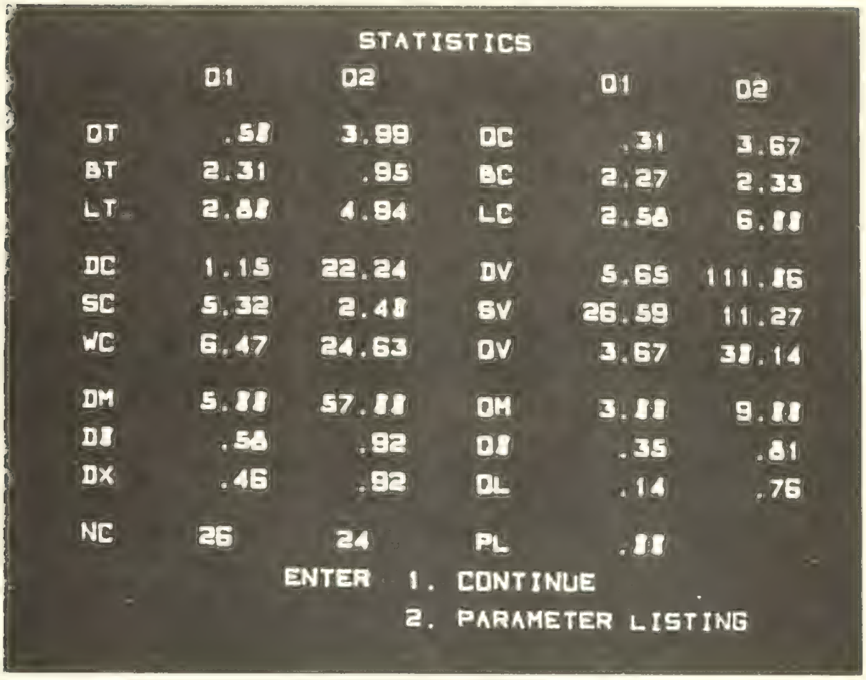
At the completion of the simulation phase, the time history is depicted dynamically on the cathode ray tube

in real time. Pictures 1 to 5 show the form of the display, with the man-like images in motion along the appropriate path, and the number in the queue box showing the queue's current length.

The timing and motion of the display are controlled by the analog computer via the hybrid connections to the digital computer. The first connection is for the purpose of timing until the next event is to be initiated. In this case, the analog computer integrates from zero to a preset time value in volts, at the rate of one volt per second. The interrupt to the appropriate subroutine then occurs, and the analog computer is reset with the next time value, after which the integration is reinitiated.

The second connection causes the dynamic motion to occur by continually checking events in progress and causing the position of the corresponding images to be incremented appropriately; this interrogation and incrementation occurs approximately 10 times per second depending on the settings of the analog computer. Movement increments are between one-half inch and two inches (depending on the path), causing a motion of between five inches and 20 inches per second.

IV. STATISTICS FROM THE SIMULATION



At the end of the simulation period, the following statistics are calculated and displayed, as shown above.

AT = time - average length of the queue

BT = time - average number of busy servers

LT = time - average number in the system = QT + BT

QC = customer - average length of the queue

BC = customer - average number of busy servers

LC = customer - average number in the system = QC + BC

DC = customer - average delay in the queue

SC = customer - average service time

WC = customer - average wait in the system = DC + SC

DM = maximum delay experienced by any customer

DO = probability that a customer's delay is greater than zero.

DX = probability that a customer's delay is greater than XD. (The value of XD is an input option.)

QM = maximum queue length reached during the simulation

QO = probability that the queue length will be greater than zero.

QL = probability that queue length will be greater than LQ (the value of LQ is an input option).

DV = standard deviation of the delay in queue

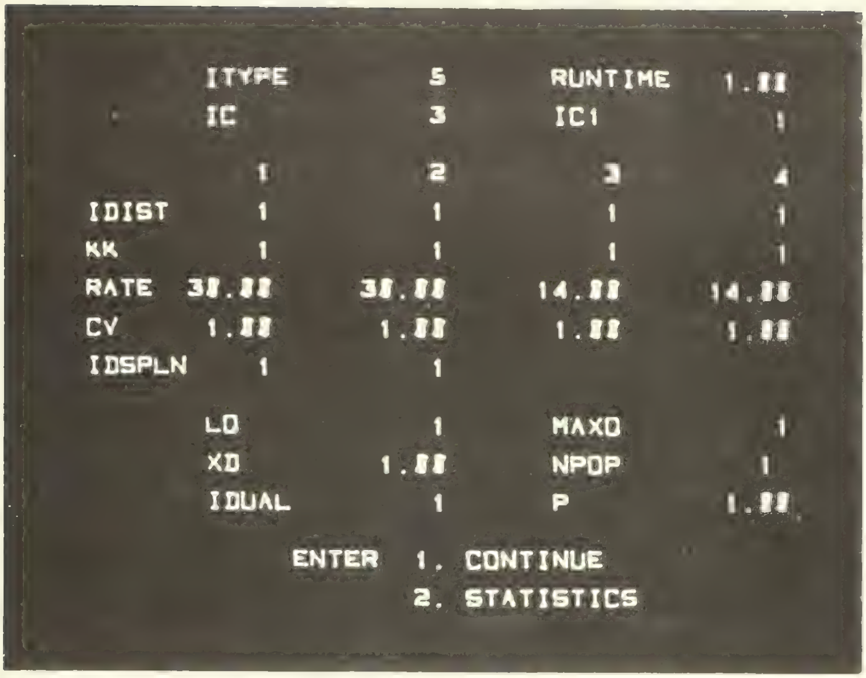
SV = standard deviation of the service times

QV = standard deviation of the queue length

NC = total number of arrivals

PS = percentage not lost (losses model only)

V. PARAMETER INPUT



Picture 7

Parameters can be input by the card reader or by the teletype terminal. In either case, the namelist form is used, where the entry form is NAME = X, NAME being the variable title and X being the value being assigned. For real variables, the decimal point must be entered. Namelist input is terminated with a *.

A. SAMPLE INPUT CARD

```
IDEV = 2, ITYPE = 3, RATE(2) = 3.0 *
```

B. NAMELIST INPUT VIA THE GRAPHICS TERMINAL

If this mode of entry is specified, one variable at a time is entered, followed by a carriage return. Entries are displayed, as entered, on the graphics screen. Termination occurs with the entry of a * followed by a carriage return.

C. INPUT VARIABLES

The following conventions are employed for specifying the options and parameters to be used.

ITYPE = 1 gives G/G/C

2 gives G/G/C with losses

3 gives G/G/C with finite source

4 gives G/G/C with feedback

5 gives two simultaneous G/G/C queues

RUNTIME = desired length of simulation (minutes)

IC = number of servers for queue 1

IC1 = number of servers for queue 2 (only applicable if ITYPE = 5 is chosen)

IDIST(I) = 1. K - ERLANG

2. HYPEREXPONENTIAL

3. DEGENERATE

KK(I) = K in K-ERLANG (use K=1 for exponential)

RATE(I) = Rate in number per minute

CV(I) = Coefficient of variation (specify for IDIST(I) = 2 only. This variable is computed otherwise.)

where I = 1 for arrivals to queue 1 (except finite model)

2 for arrivals to queue 2 or finite population idle parameters

3 for service for queue 1 (except finite model)

4 for service for queue 2 or finite population service parameters

IDSPLN(I) = 1. FIFO

2. LIFO

3. Shortest service time first

4. Random draw

where I = 1 for queue 1

2 for queue 2

LQ = parameter in the statistic P(Q > LQ)

XD = parameter in the statistic P(wait > XD)

IDUAL = 0. Independent arrival streams

1. Identical arrival streams

(applicable to ITYPE = 5 only)

MAXQ = maximum queue size (for G/G/C with losses only)

NPOP = population size (for finite source option only)

P = completion probability (for feedback option only)

VI. THE PROGRAM

The program requires the use of the XDS 9300 digital computer, the CI 5000 analog computer, and a graphics terminal using the GATED monitor.

The analog computer wiring diagram necessary to drive the display sequence appears in APPENDIX A. Before beginning, set Delay Flop Switches DF00, DF01, and DF02 to .1 MS and set counter F00 to 1 and F04 to 0. Then select RUN and DIGITAL COMPUTER on the analog control panel. This provides for movement interrupts to occur at a rate of 100/11 hertz. If desired, the timing values will be displayed on the RATIOMETER by selecting TRUNK 420 prior to DIGITAL COMPUTER.

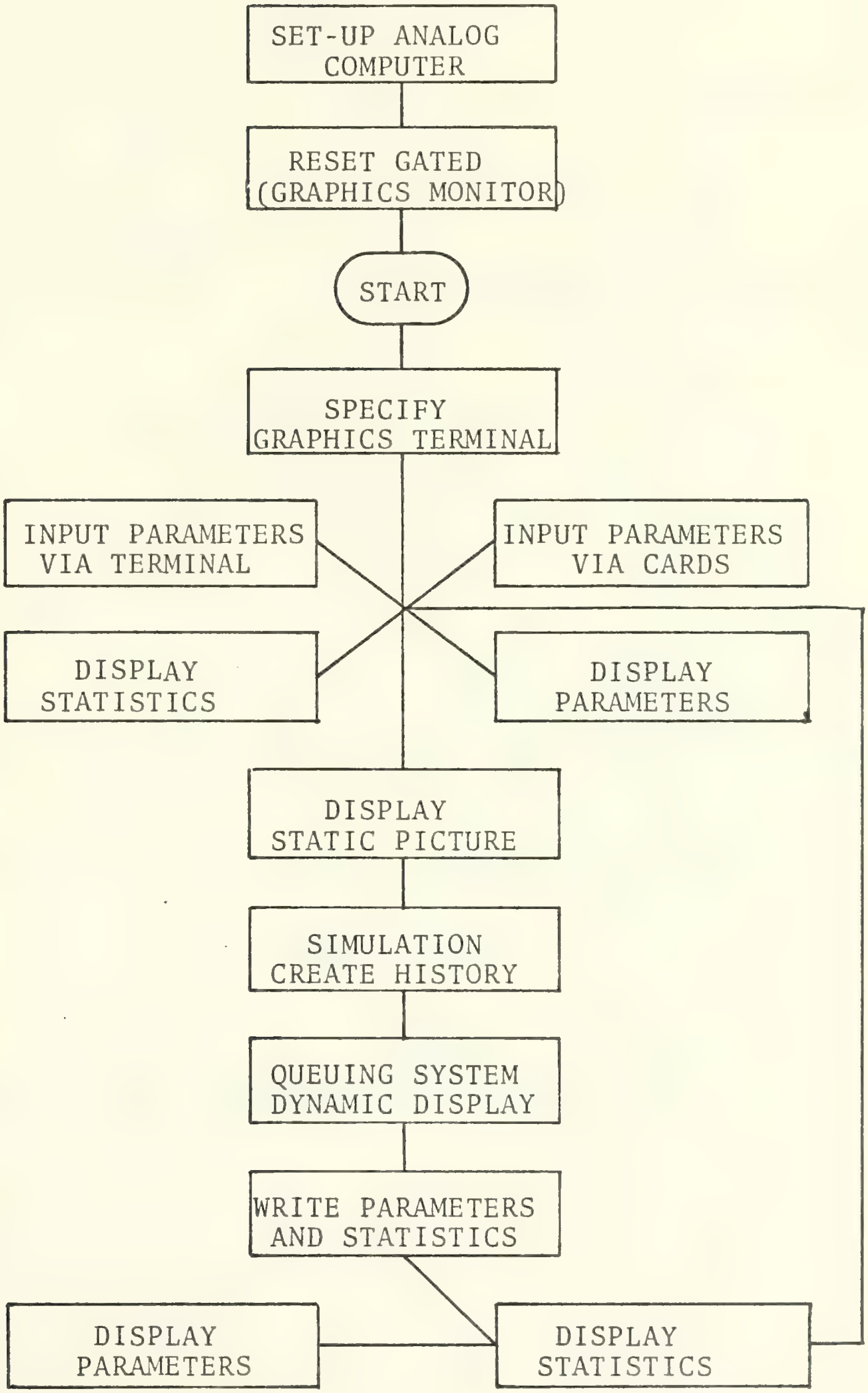
The program consists of a FORTRAN program and a META-SYMBOL 9300 program. Because the history is buffered out to the device assigned the logical unit value 7, an additional control card is needed.

Job cards for deck run should be:

```
¬JOB
¬AGT
¬ASSIGN 7=DF1A
¬FORTRAN LS,GO

 FORTRAN DECK

¬META 9300 SI,LO,GO

 META-SYMBOL DECK
```

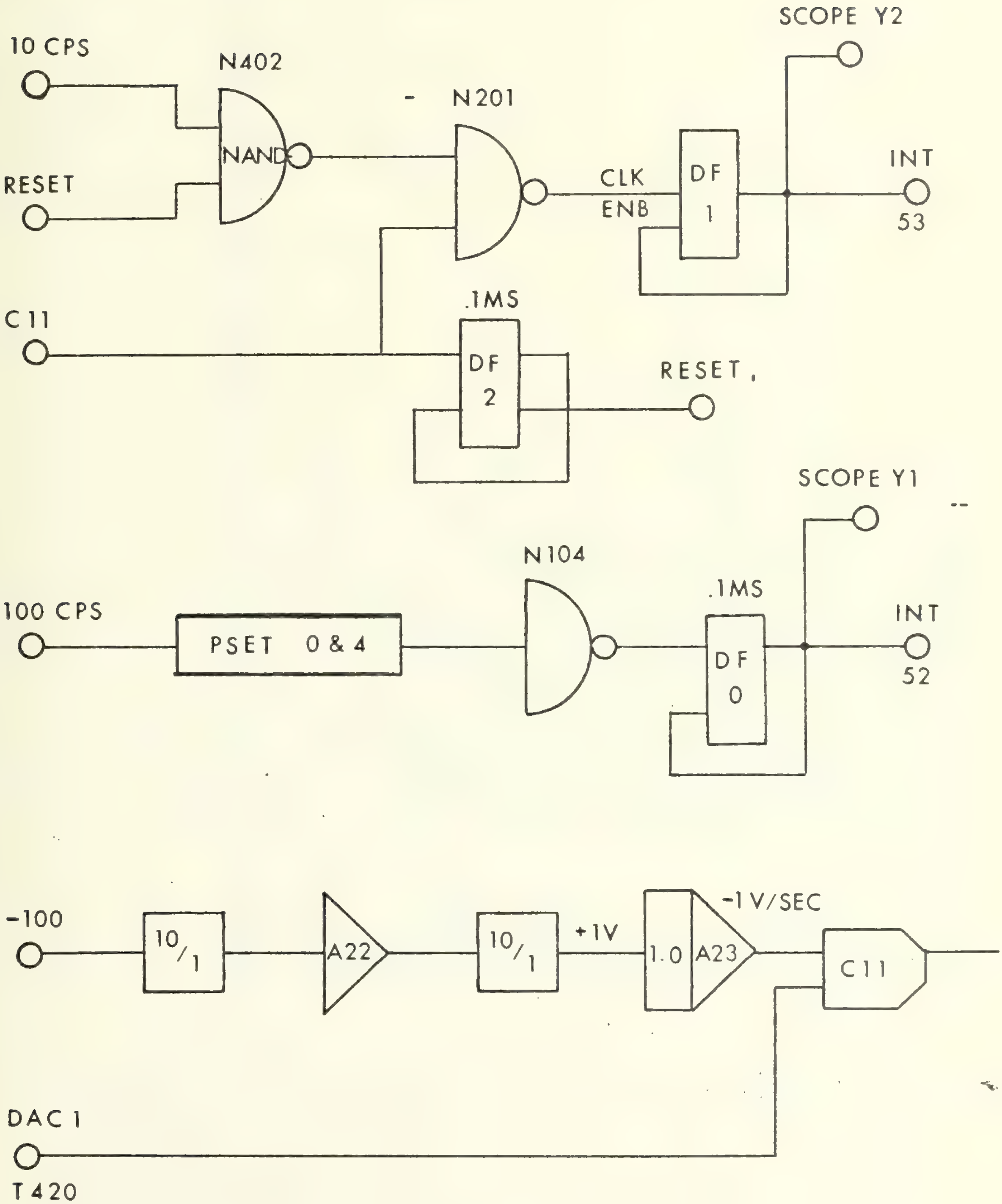
```
¬EOF
¬LOAD XR,MAP
¬DATA
```

PROGRAM STEPS



APPENDIX A

ANALOG SCHEMATIC



```
C     ************************************************************
C     ************************************************************
C
C
C     ***  SIMULATION OF QUEUEING MODELS WITH DYNAMIC DISPLAY  ***
C
C
C                           C. M. RIDDELL
C
C                            MARCH 1973
C
C
C     ************************************************************
C     ************************************************************
C
C
C     DICTIONARY OF EXOGENOUS VARIABLES
C
C
C     CHNG     - INCREMENT VALUES
C     CLOCK    - HISTORY TIMES
C     CLTIME   - SERVICE TIME SUM
C     CLTIMESQ- (SERVICE TIME)**2 SUM
C     CTIMSQ   - (QUEUE LENGTH)**2 SUM
C     CUSTIM   - QUEUE LENGTH SUM
C     CV       - COEFICIENT OF VARIATION
C     DELAY    - QUEUE WAIT SUM
C     DELAYO   - DELAYS GREATER THAN 0
C     DELAYSQ - (QUEUE WAIT)**2 SUM
C     DELAYX   - DELAYS GREATER THAN XD
C     ENDD     - MOVEMENT END VALUES
C     FTIME    - NEXT EVENT TIMES
C     I3       - HISTORY POINTER
C     IBUFF    - HISTORY BUFFER
C     IBUSY    - BUSY?  0 - NO:  1 - YES:  2 - NOT APPLICABLE
C     IC       - NUMBER OF SERVERS QUEUE 1
C     IC1      - NUMBER OF SERVERS QUEUE 2
C     IDEV     - SPECIFIES GRAPHICS TERMINAL
C     IDIST    - DISTRIBUTION TYPE
C     IDSPLN   - QUEUE DISCIPLINE
C     IDUAL    - IDENTICAL ARRIVALS?  0 - NO:  1 - YES
C     IFDBK    - FEEDBACK?  0 - NO:  1 - YES
C     IFLAG    - DYNAMIC IMAGE FLAGS
C     ILN      - TEXT LINE NUMBER
C     IMAN     - DYNAMIC IMAGES
C     IMD      - MOVE-DRAW INDICATORS
C     IPOS     - TEXT STARTING POSITION
C     ISTOT    - ARRIVAL BUSY SERVER SUM
```

```
C      ITOT     - INTERMEDIATE HISTORY COUNTER
C      ITPOP    - POPULATION SIZE TEXT WORD
C      ITQ      - QUEUE LENGTH TEXT WORDS
C      ITXA     - MAIN CHOICE PAGE TEXT WORD
C      ITYPE    - MODEL TYPE
C      IX1-IX4  - RANDOM NUMBER SEEDS
C      KIM      - HISTORY IMAGE POINTER
C      KK       - K (ERLANG)
C      KP       - VECTOR POINTERS FOR TIM
C      LINE     - LENGTH OF QUEUES
C      LOG      - HISTORY LOGIC POINTER
C      LQ       - LOWER LIMIT QUEUE LENGTH VALUE
C      LTOT     - ARRIVAL QUEUE SUM
C      MAXLN    - MAXIMUM QUEUE VALUE REACHED
C      MAXQ     - DESIRED MAX QUEUE LENGTH
C      MD       - MOVE-DRAW INDICATORS
C      XMDELAY  - MAXIMUM DELAY IN QUEUE
C      NCUST    - ARRIVAL COUNTERS
C      NFDBK    - TOTAL NUMBER OF FEEDBACKS
C      NPOP     - FINITE POPULATION SIZE
C      NQ       - LENGTH OF QUEUES
C      NSIM     - NUMBER OF STATIC IMAGES
C      NTOT     - TOTAL NUMBER OF HISTORY ENTRIES
C      NULL     - BLANK
C      P        - PROBABILITY
C      PROBO    - QUEUE GREATER THAN O SUM
C      PROBL    - QUEUE GREATER THAN LQ SUM
C      PSTIME   - PREVIOUS STATE CHANGE TIME
C      PTIME    - PREVIOUS QUEUE CHANGE TIME
C      RATE     - RATE
C      RUNTIME  - LENGTH OF SIMULATION
C      TIM      - STORED TIMES
C      TIME     - CURRENT TIME
C      XD       - LOWER LIMIT TIME VALUE
C      X        - SCREEN X COORDINATES
C      XX       - DYNAMIC IMAGE X COORDINATES
C      XXX      - DYNAMIC IMAGE STARTING POSITIONS
C      Y        - SCREEN Y COORDINATES
C      YY       - DYNAMIC IMAGE Y COORDINATES
C      YYY      - DYNAMIC IMAGE STARTING POSITIONS
C
C      ****************************************************************
```

```
      DIMENSION IGDIR(30),ITDIR(16)
      DIMENSION IBUFF(400)
      DIMENSION ITXA(6)
      REAL LT,LC
      COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
      COMMON /SET5/ IDEV,IER
      COMMON /SET6/ CLOCK(100),KIM(100),LOG(100)
      COMMON /SET8/ CHNG(10),ENDD(5,22),IFLAG(22),ITOT,IPOS(2),
     -ILN(2),IMAN(7,22),NSIM,NQ(2),ITQ(2,2),NPOP,ITPOP(2),XX(5,22),
     -XXX(5,22),YY(5,22),YYY(5,22),IMD(5)
      COMMON /SET9/ IDIST(4),RATE(4),KK(4),CV(4),NFDBK,IFDBK,IDSPLN(2),
     -IDUAL,IBUSY(6),KP(2),LINE(2),TIME,PSTIME,PTIME(2),RUNTIME,IX1,
     -IX2,IX3,IX4,XD,LQ,MAXQ,P
      COMMON /SET10/ NTOT,ITYPE
      COMMON /SET11/ NCUST(2),MAXLN(2),DELAY(2),DELAYSQ(2),XMDELAY(2),
     -DELAYO(2),DELAYX(2),CLTIME(2),CLTIMSQ(2),CUSTIM(2),CTIMSQ(2),
     -PROBO(2),PROBL(2),LTOT(2),ISTOT(2)
      COMMON /STAT/ QT(2),BT(2),LT(2),QC(2),BC(2),LC(2),
     -DC(2),SC(2),WC(2),DM(2),DO(2),DX(2),QM(2),QO(2),
     -QL(2),DV(2),SV(2),QV(2),PL,NC(2)
      COMMON /SETX/ A1,A2,A3,A4,G1,G2,G3,G4,S1,S2,S3,S4,S5,S6,S7,S8
      EQUIVALENCE (IBUFF(1),CLOCK(1)),(IBUFF(201),KIM(1)),
     -(IBUFF(301),LOG(1))
      DATA ITQ(2,1),ITQ(2,2),ITPOP(2),NULL/77777777B/
      DATA IX1,IX2,IX3,IX4/55403,76941,10799,33999/
      DATA X/-1.08,-.94,-.81,-.74,-.66,-.58,-.51,-.26,-.15,.15,.58,
     -.74,.94,1.34/
      DATA Y/.57,.48,.4,.36,.28,.2,.16,.13,.08,.06,.0,-.02,-.04,
     --.1,-.12,-.17,-.2,-.24,-.32,-.4,-.44,-.52,-.6,-.72,-1.0/
      DATA ITYPE,IC,IC1,IDIST,KK,IDSPLN,LQ,IDUAL,MAXQ,NPOP/1/
      DATA RATE,CV,RUNTIME,XD,P/1.0/
      DATA MD/6*0/, IMD/0,4*1/
C
C     ESTABLISH NAMELIST VARIABLES (FOR INPUT)
C
      NAMELIST ITYPE,IC,IC1,IDIST,KK,RATE,CV,IDSPLN,RUNTIME,
     -LQ,XD,IDUAL,MAXQ,NPOP,P,IDEV,IPRINT
C
C     FORMATS FOR MAIN CHOICE PAGE
C
  101 FORMAT('1.  INPUT BY CARDS      ')
  102 FORMAT('2.  INPUT BY TERMINAL   ')
  103 FORMAT('ENTER CHOICE THEN RETURN')
  104 FORMAT('3.  PARAMETER LISTING   ')
  105 FORMAT('4.  RUN',17X)
C
C     MAKE THE HYBRID CONNECTIONS
C
```

```
C     52 INTERUPT CAUSES MOTION
C     53 INTERUPT TIMES EVENTS
C
      CONNECT (52,FLAGS)
      CONNECT (53,SETFLAGS)
C
C     DEFAULTS: NO HISTORY OUTPUT: GRAPHICS TERMINAL 1
C
      IPRINT=0: IDEV=1
      OUTPUT(101) 'TYPE IDEV=2 * AND A C/R IF AGT-2 IS TO BE USED ELSE T
     -YPE * AND A C/R'
      INPUT(101)
C
C     SET THE STATIC IMAGE COORDINATES
C
      CALL PACK
C
C     INITIALIZE THE GRAPHICS AND TEXT DIRECTORIES
C
      CALL DGINIT (IDEV,IGDIR,30,IER)
   99 CALL DTINIT (IDEV,ITDIR,16,IER)
C
C     PRESENT MAIN CHOICE PAGE
C
  110 ENCODE(24,101,ITXA)
      CALL TEXTO (IDEV,ITXA,6,15,25,2,3,IER)
      ENCODE(24,102,ITXA)
      CALL TEXTO (IDEV,ITXA,6,17,25,2,3,IER)
      ENCODE(24,104,ITXA)
      CALL TEXTO (IDEV,ITXA,6,19,25,2,3,IER)
      ENCODE(24,105,ITXA)
      CALL TEXTO (IDEV,ITXA,6,21,25,2,3,IER)
      ENCODE(24,103,ITXA)
      CALL TEXTO (IDEV,ITXA,6,24,25,2,3,IER)
      CALL TEXTR (IDEV,NULL,1,26,47,2,3,IER)
C
C     WAIT FOR REPLY
C
  100 IF(MOD(ITDIR(6),8).EQ.0) GO TO 100
      CALL TEXTI (IDEV,I,1,26,47,IER)
C
C     LOGICAL RIGHT SHIFT 18 BITS (3 CHARACTERS)  DECODES REPLY
C
      I=LRS(I,18)
C
      CALL DTINIT(IDEV,ITDIR,16,IER)
C
C     GET CARD INPUT
```

```
C
      IF(I.NE.1) GO TO 113
      INPUT(5)
      DO 114 J=1,4
      IF(IDIST(J).EQ.3) CV(J)=0.0
  114 IF(IDIST(J).EQ.1) CV(J)=1.0/SQRT(KK(J))
      GO TO 110
  113 IF(I.NE.3) GO TO 112
      IF(I.NE.3) GO TO 112
  111 CALL PARAMETERS
      CALL TEXTR (IDEV,NULL,1,34,9,2,3,IER)
  120 IF(MOD(ITDIR(14),8).EQ.0) GO TO 120
      CALL TEXTI (IDEV,I,1,34,9,IER)
      CALL DTINIT (IDEV,ITDIR,16,IER)
      I=LRS(I,18)
      IF(I.EQ.1) GO TO 110
      IF(I.NE.2) GO TO 111
  121 CALL STATISTICS
      CALL TEXTR (IDEV,NULL,1,35,11,2,3,IER)
  125 IF(MOD(ITDIR(15),8).EQ.0) GO TO 125
      CALL TEXTI (IDEV,I,1,35,11,IER)
      CALL DTINIT (IDEV,ITDIR,16,IER)
      I=LRS(I,18)
      IF(I.EQ.1) GO TO 110
      IF(I.EQ.2) GO TO 111
      GO TO 121
  112 IF(I.EQ.4) GO TO 1
      IF(I.NE.2) GO TO 110
      IBLK=1
C
C     GET TERMINAL INPUT
C
      CALL GINPUT (IDEV,ITDIR,IBLK)
      DO 115 J=1,4
      IF(IDIST(J).EQ.3) CV(J)=0.0
  115 IF(IDIST(J).EQ.1) CV(J)=1.0/SQRT(KK(J))
      GO TO 99
C
C     CONVERT MINUTES TO SECONDS
C
    1 DO 2 I=1,4
    2 RATE(I)=RATE(I)/60.
      RUNTIME=RUNTIME*60
C
C     BEGIN MODEL INITIALIZATION OF VALUES
C
      IF(ITYPE.GE.5) GO TO 4
      DO 3 I=1,6
```

```
      IF(I.LE.IC) IBUSY(I)=0: GO TO 3
      IBUSY(I)=2
    3 CONTINUE
      GO TO 7
    4 DO 6 I=1,3
      IF(I.LE.IC) IBUSY(I)=0: GO TO 5
      IBUSY(I)=2
    5 IF(I.LE.IC1) IBUSY(I+3)=0: GO TO 6
      IBUSY(I+3)=2
    6 CONTINUE
    7 NFDBK=IFDBK=ITOT=NTOT=TIME=PSTIME=0
      DO 8 I=1,22
    8 IFLAG(I)=0
      DO 15 I=1,2
   15 KP(I)=LINE(I)=NQ(I)=NCUST(I)=MAXLN(I)=DELAY(I)=DELAYSQ(I)=
     -XMDELAY(I)=DELAYO(I)=DELAYX(I)=CLTIME(I)=CLTIMSQ(I)=
     -PTIMF(I)=CUSTIM(I)=CTIMSQ(I)=PROBO(I)=PROBL(I)=
     -LTOT(I)=ISTOT(I)=0
      IF(ITYPE.EQ.5) GO TO 50
    9 A1=A2=1: A3=A4=12
      G1=G2=10: G3=G4=12
      S1=S2=12: S3=15: S4=9: S5=19: S6=5: S7=22: S8=2
      CHNG(5)=-.0333: CHNG(6)=.0333: CHNG(7)=-.1
      CHNG(8)=.1: CHNG(9)=-.1666: CHNG(10)=.1666
      ILN(1)=ILN(2)=22: IPOS(1)=41: IPOS(2)=16
      GO TO (10,20,30,40) ITYPE
C
C     SETUP G/G/C
C
   10 CALL GGC
      DO 11 I=1,4
   11 ENDD(5,I)=X(9)
      DO 12 I=5,10
   12 ENDD(1,I)=X(12)
      DO 13 I=11,22
   13 ENDD(5,I)=X(14)+.1
      CHNG(1)=.225: CHNG(2)=.17: CHNG(3)=.2
      NSIM=5
      GO TO 85
C
C     SETUP G/G/C WITH LOSSES
C
   20 CALL LOSSES
      ENDD(5,1)=ENDD(5,3)=ENDD(1,2)=ENDD(1,4)=X(9)
      ENDD(2,2)=ENDD(2,4)=Y(25)
      DO 22 I=5,10
   22 ENDD(1,I)=X(12)
      DO 21 I=11,22
```

```
   21 ENDD(5,I)=X(14)+.1
      CHNG(1)=.225; CHNG(2)=.17; CHNG(3)=.205
      NSIM=6
      GO TO 85
C
C     SETUP G/G/C WITH FINITE SOURCE
C
   30 CALL FINITE
      A1=A2=7
      DO 31 I=1,4
   31 ENDD(5,I)=X(9)+.03
      DO 32 I=5,10
   32 ENDD(1,I)=X(12)
      DO 33 I=11,22
      ENDD(1,I)=X(13)+.1
      ENDD(2,I)=Y(24)-.1
      ENDD(3,I)=X(5)
   33 ENDD(4,I)=Y(16)
      CHNG(1)=.12; CHNG(2)=.17; CHNG(3)=.2; CHNG(4)=.275
      NSIM=7
      GO TO 85
C
C     SETUP G/G/C WITH FEEDBACK TO THE QUEUE
C
   40 CALL FEEDBK
      DO 41 I=1,4
   41 ENDD(5,I)=X(9)
      DO 42 I=5,10
   42 ENDD(1,I)=X(12)
      DO 43 I=11,22
      ENDD(1,I)=X(13)+.1
      ENDD(2,I)=Y(24)-.1
      ENDD(3,I)=X(8)
      ENDD(4,I)=Y(12)
   43 ENDD(5,I)=X(14)+.1
      CHNG(1)=.225; CHNG(2)=.17; CHNG(3)=.2; CHNG(4)=.35
      NSIM=7
      GO TO 85
C
C     SETUP TWO G/G/C QUEUES
C
   50 CALL TWOGGC
      ILN(1)=17; ILN(2)=27; IPOS(1)=IPOS(2)=41
      A1=A2=1; A3=5; A4=19
      G1=G2=10; G3=5; G4=19
      S1=S2=12; S3=5; S4=9; S5=2; S6=19; S7=22; S8=15
      DO 51 I=1,4
   51 ENDD(5,I)=X(9)
```

```
      DO 52 I=5,10
   52 ENDD(1,I)=X(12)
      DO 53 I=11,22
   53 ENDD(5,I)=X(14)+.1
      CHNG(1)=.225; CHNG(2)=.17; CHNG(3)=.2
      CHNG(5)=CHNG(8)=0.0
      CHNG(6)=CHNG(9)=-.0666; CHNG(7)=CHNG(10)=.0666
      NSIM=6
C
C     SET COORDINATES FOR DYNAMIC IMAGES
C
   85 CALL XFS
C
C     BEGIN THE SIMULATION
C
      CALL SIMULATE
C
C     PRINT THE HISTORY?
C
      IF(IPRINT.EQ.0) GO TO 90
      DO 86 J=1,IPRINT
   86 OUTPUT(6) CLOCK(J),KIM(J),LOG(J)
C
C     BEGIN THE DYNAMIC GRAPHICS DISPLAY
C
   90 CALL DRIVER
C
C     LIST STATISTICS
C
      OUTPUT(6) ITYPE,RUNTIME,IC,IC1,IDIST,KK,RATE,CV,
     -IDSPLN,LQ,XD,IDUAL,MAXQ,NPOP,P
      OUTPUT(6) QT,BT,LT,QC,BC,LC,DC,SC,WC,DM,DO,DX,QM,QO,QL,
     -DV,SV,QV,PL,NC
      CALL DGINIT (IDEV,IGDIR,30,IER)
      CALL DTINIT(IDEV,ITDIR,16,IER)
C
C     CONVERT SECONDS TO MINUTES
C
      DO 95 I=1,4
   95 RATE(I)=RATE(I)*60.
      RUNTIME=RUNTIME/60.
C
C     DISPLAY STATISTICS
C
   91 CALL STATISTICS
      CALL TEXTR (IDEV,NULL,1,35,11,2,3,IER)
  150 IF(MOD(ITDIR(15),8).EQ.0) GO TO 150
      CALL TEXTI (IDEV,I,1,35,11,IER)
```

```
      CALL DTINIT(IDEV,ITDIR,16,IER)
      I=LRS(I,18)
      IF(I.EQ.1) GO TO 110
      IF(I.NE.2) GO TO 91
C
C     DISPLAY PARAMETERS?
C
   93 CALL PARAMETERS
      CALL TEXTR (IDEV,NULL,1,34,9,2,3,IER)
  175 IF(MOD(ITDIR(14),8).EQ.0) GO TO 175
      CALL TEXTI (IDEV,I,1,34,9,IER)
      CALL DTINIT(IDEV,ITDIR,16,IER)
      I=LRS(I,18)
      IF(I.EQ.2) GO TO 91
C
C     RETURN FOR ANOTHER RUN
C
      GO TO 110
      END
```

```
C         ******************************************************************
C
C         SUBROUTINE SIMULATE
C
C         PURPOSE: TO CREATE THE TIME HISTORY OF EVENTS
C                  AND GATHER STATISTICS
C
C         SUBROUTINES USED:  ARRIV, BEGSRV, ENDSRV, ACCUM, TIMES
C
C         ******************************************************************
C

          SUBROUTINE SIMULATE
          DIMENSION IBUFF(400)
          REAL LT,LC
          COMMON /SET6/ CLOCK(100),KIM(100),LOG(100)
          COMMON /SET7/ ETIME(9),I,I3,K1,TIM(115,4)
          COMMON /SET8/ CHNG(10),ENDD(5,22),IFLAG(22),ITOT,IPOS(2),
         -ILN(2),IMAN(7,22),NSIM,NQ(2),ITQ(2,2),NPOP,ITPOP(2),XX(5,22),
         -XXX(5,22),YY(5,22),YYY(5,22),IMD(5)
          COMMON /SET9/ IDIST(4),RATE(4),KK(4),CV(4),NFDBK,IFDBK,IDSPLN(2),
         -IDUAL,IBUSY(6),KP(2),LINE(2),TIME,PSTIME,PTIME(2),RUNTIME,IX1,
         -IX2,IX3,IX4,XD,LQ,MAXQ,P
          COMMON /SET10/ NTOT,ITYPE
          COMMON /SET11/ NCUST(2),MAXLN(2),DELAY(2),DELAYSQ(2),XMDELAY(2),
         -DELAYO(2),DELAYX(2),CLTIME(2),CLTIMSQ(2),CUSTIM(2),CTIMSQ(2),
         -PROBO(2),PROBL(2),LTOT(2),ISTOT(2)
          COMMON /STAT/ QT(2),BT(2),LT(2),QC(2),BC(2),LC(2),
         -DC(2),SC(2),WC(2),DM(2),DO(2),DX(2),QM(2),QO(2),
         -QL(2),DV(2),SV(2),QV(2),PL,NC(2)
          EQUIVALENCE (IBUFF(1),CLOCK(1)),(IBUFF(201),KIM(1)),
         -(IBUFF(301),LOG(1))
          I3=1
C
C         SET EVENT TIMES TO LARGE VALUE
C
          DO 10 K=1,8
       10 ETIME(K)=99999.
          IF(ITYPE.NE.3) GO TO 4
          I=2
C
C         IDLE TIMES FOR FINITE POPULATION
C
          DO 3 J=1,NPOP
        3 CALL TIMES
          TIM(KP(1)+1,1)=TIM(KP(2),2)
          TIM(KP(1)+1,3)=TIM(KP(2),4)
```

```
      KP(2)=KP(2)-1
C
C     SET FIRST ARRIVAL FROM FINITE SOURCE
C
      ETIME(7)=TIM(KP(1)+1,1)
      GO TO 5
C
C     GET FIRST ARRIVAL TIME
C
    4 I=1; CALL TIMES; ETIME(7)=TIM(1,1)
      IF(ITYPE.EQ.5) I=2; CALL TIMES; ETIME(8)=TIM(1,2)
C
C     SET LENGTH OF SIMULATION
C
    5 ETIME(9)=RUNTIME
C
C     DETERMINE NEXT EVENT
C
   11 TIME=ETIME(1)
      K1=1
      DO 15 K=2,9
      IF(ETIME(K).LT.TIME) TIME=ETIME(K); K1=K
   15 CONTINUE
C
C     SEPARATE END-OF-SERVICE
C
      IF(K1.GT.6) GO TO 20
      I=1
      IF(ITYPE.GE.5 .AND. K1.GE.4) I=2
      CALL ENDSRV; GO TO 11
C
C     SEPARATE ARRIVALS
C
   20 IF(K1.LE.8) I=K1-6; CALL ARRIV; GO TO 11
C
C     DONE: COMPLETE HISTORY
C
      CALL BUFFEROUT(7,1,IBUFF,400,IND)
   21 IF(IND.EQ.1) GO TO 21
      GO TO (21,23,22,22) IND
   22 OUTPUT(101) 'BUFFERING ERROR 5'
   23 REWIND 7
      I=1
C
C     COMPLETE TIME STATISTICS
C
   50 CALL ACCUM
C
```

```
C     CALCULATE FINAL STATISTICAL VALUES
C
      TCUST=NCUST(I)
      IF(ITYPE.EQ.2) TCUST=NCUST(1)+NCUST(2);
     -PL=NCUST(1)/TCUST; NCUST(1)=TCUST
      DO 55 J=1,KP(I)
      WAIT=TIME-TIM(J,I)
      DELAY(I)=DELAY(I)+WAIT
      IF(WAIT.GT.XMDELAY(I)) XMDELAY(I)=WAIT
      DELAYO(I)=DELAYO(I)+1
   55 IF(WAIT.GT.XD) DELAYX(I)=DELAYX(I)+1
      SC(I)=CLTIME(I)/(TCUST-LINE(I))
      SV(I)=SQRT(CLTIMSQ(I)/(TCUST-LINE(I))-SC(I)**2)
      K=1; M1=1
      IF(ITYPE.NE.5) M2=6; GO TO 59
      IF(I.EQ.1) M2=3; GO TO 59
      M1=4; M2=6; K=2
   59 DO 60 J=M1,M2
      IF(ETIME(J).EQ.99999.) GO TO 60
      CLTIME(K)=CLTIME(K)-(ETIME(J)-TIME)
   60 CONTINUE
      QT(I)=CUSTIM(I)/TIME
      BT(I)=CLTIME(I)/TIME
      LT(I)=QT(I)+BT(I)
      QC(I)=LTOT(I)/TCUST
      BC(I)=ISTOT(I)/TCUST
      LC(I)=QC(I)+BC(I)
      DC(I)=DELAY(I)/TCUST
      WC(I)=DC(I)+SC(I)
      DM(I)=XMDELAY(I)
      DO(I)=DELAYO(I)/TCUST
      DX(I)=DELAYX(I)/TCUST
      QM(I)=MAXLN(I)
      QO(I)=PROBO(I)/TIME
      QL(I)=PROBL(I)/TIME
      DV(I)=SQRT(DELAYSQ(I)/TCUST-DC(I)**2)
      QV(I)=SQRT(CTIMSQ(I)/TIME-QT(I)**2)
      NC(I)=NCUST(I)
      IF(ITYPE.NE.2) PL=0.0
      IF(ITYPE.NE.5) GO TO 70
      IF(I.EQ.1) I=2; GO TO 50
      GO TO 80
   70 QT(2)=BT(2)=LT(2)=QC(2)=BC(2)=LC(2)=DC(2)=SC(2)=
     -WC(2)=DV(2)=SV(2)=QV(2)=DM(2)=DO(2)=DX(2)=QM(2)=
     -QO(2)=QL(2)=NC(2)=0
   80 RETURN
      END
```

```
C     ****************************************************************
C
C     SUBROUTINE ARRIV
C
C     PURPOSE:  RECORD AN ARRIVAL TO THE QUEUE IN THE HISTORY,
C               GATHER STATISTICS,  AND DETERMINE IF A SERVER
C               IS AVAILABLE
C
C     ****************************************************************
C

      SUBROUTINE ARRIV
      DIMENSION IBUFF(400)
      DIMENSION NO(2)
      DATA NO/1,2/
      COMMON /SET6/ CLOCK(100),KIM(100),LOG(100)
      COMMON /SET7/ ETIME(9),I,I3,K1,TIM(115,4)
      COMMON /SET9/ IDIST(4),RATE(4),KK(4),CV(4),NFDBK,IFDBK, IDSPLN(2),
     -IDUAL,IBUSY(6),KP(2),LINE(2),TIME,PSTIME,PTIME(2),RUNTIME,IX1,
     -IX2,IX3,IX4,XD,LQ,MAXQ,P
      COMMON /SET10/ NTOT,ITYPE
      COMMON /SET11/ NCUST(2),MAXLN(2),DELAY(2),DELAYSQ(2),XMDELAY(2),
     -DELAYO(2),DELAYX(2),CLTIME(2),CLTIMSQ(2),CUSTIM(2),CTIMSQ(2),
     -PROBO(2),PROBL(2),LTOT(2),ISTOT(2)
      EQUIVALENCE (IBUFF(1),CLOCK(1)),(IBUFF(201),KIM(1)),
     -(IBUFF(301),LOG(1))
      KP(I)=KP(I)+1
C
C     QUEUE AT CAPACITY? STOP
C
      IF(KP(I).GE.114) ETIME(9)=TIME-1.0; RETURN
C
C     GATHER STATISTICS
C
      LTOT(I)=LTOT(I)+LINE(I)
      IF(ITYPE.EQ.5) GO TO 51
      DO 50 J=1,6
      IF(IBUSY(J).EQ.2) GO TO 54
   50 ISTOT(1)=ISTOT(1)+IBUSY(J)
      GO TO 54
   51 DO 52 J=1,3
      IF(IBUSY(J).EQ.2) GO TO 54
   52 ISTOT(I)=ISTOT(I)+IBUSY(J)
   54 IF(IFDBK.EQ.1) GO TO 13
      IF(ITYPE.NE.2) GO TO 5
C
C     DETERMINE IF LOST
```

```
C
      IF(LINE(1).LT.MAXQ) GO TO 5
C
C     HISTORY ENTRY
C
      CLOCK(I3)=TIME-PSTIME
      KIM(I3)=NO(2)
C
C     ALTERNATE IMAGES
C
      IF(NO(2).EQ.2) NO(2)=4; GO TO 101
      NO(2)=2
  101 LOG(I3)=2
      I3=I3+1; NTOT=NTOT+1
      IF(I3.NE.101) GO TO 4
      CALL BUFFEROUT(7,1,IBUFF,400,IND)
    1 IF(IND.EQ.1) GO TO 1
      GO TO (1,3,2,2) IND
    2 OUTPUT(101) 'BUFFERING ERROR 1'
    3 I3=1
    4 NCUST(2)=NCUST(2)+1
      PSTIME=TIME
      KP(1)=KP(1)-1
      GO TO 40
    5 IF(ITYPE.GE.5 .OR. ITYPE.EQ.2) J=I; GO TO 6
      IF(J.EQ.1) J=2; GO TO 6
      J=1
C
C     ALTERNATE IMAGES
C
    6 IF(NO(J).EQ.J) NO(J)=J+2; GO TO 8
      NO(J)=J
C
C     HISTORY ENTRY
C
    8 KIM(I3)=NO(J)
      CLOCK(I3)=TIME-PSTIME
      LOG(I3)=1
      I3=I3+1; NTOT=NTOT+1
      IF(I3.NE.101) GO TO 12
      CALL BUFFEROUT(7,1,IBUFF,400,IND)
    9 IF(IND.EQ.1) GO TO 9
      GO TO (9,11,10,10) IND
   10 OUTPUT(101) 'BUFFERING ERROR 2'
   11 I3=1
C
C     TIME STATISTICS
C
```

```
   13 NCUST(I)=NCUST(I)+1
      LINE(I)=LINE(I)+1
      IF(IFDBK.EQ.1) IFDBK=0; GO TO 42
C
C     SERVER AVAILABLE?
C
      IF(LINE(I).GT.1) GO TO 40
      LOG(I3)=7
      IF(ITYPE.LE.4) GO TO 30
      IF(I.EQ.2) GO TO 20
      DO 15 K1=1,3
   15 IF(IBUSY(K1).EQ.0) CALL BEGSRV; GO TO 40
      GO TO 40
   20 DO 21 K1=4,6
   21 IF(IBUSY(K1).EQ.0) CALL BEGSRV; GO TO 40
      GO TO 40
   30 DO 31 K1=1,6
   31 IF(IBUSY(K1).EQ.0) CALL BEGSRV; GO TO 40
   40 IF(ITYPE.NE.3) GO TO 45
C
C     FINITE SOURCE EMPTY?
C
      IF(KP(2).EQ.0) ETIME(7)=99999.; GO TO 42
C
C     SWITCH NEXT ARRIVAL TO QUEUE HALF OF ARRAY
C
      TIM(KP(1)+1,1)=TIM(KP(2),2)
      TIM(KP(1)+1,3)=TIM(KP(2),4)
      KP(2)=KP(2)-1
      GO TO 41
C
C     GET NEXT ARRIVAL TIMES
C
   45 CALL TIMES
   41 ETIME(I+6)=TIM(KP(I)+1,I)
   42 IF(LINE(I).GT.MAXLN(I)) MAXLN(I)=LINE(I)
      RETURN
      END
```

```
C     ****************************************************************
C
C     SUBROUTINE BEGSRV
C
C     PURPOSE:  RECORD BEGINNING OF SERVICE;  SELECTS QUEUE MEMBER
C               AND SETS END OF SERVICE TIME
C
C     ****************************************************************
C
      SUBROUTINE BEGSRV
      DIMENSION IBUFF(400)
      COMMON /SET6/ CLOCK(100),KIM(100),LOG(100)
      COMMON /SET7/ ETIME(9),I,I3,K1,TIM(115,4)
      COMMON /SET9/ IDIST(4),RATE(4),KK(4),CV(4),NFDBK,IFDBK,IDSPLN(2),
     -IDUAL,IBUSY(6),KP(2),LINE(2),TIME,PSTIME,PTIME(2),RUNTIME,IX1,
     -IX2,IX3,IX4,XD,LQ,MAXQ,P
      COMMON /SET10/ NTOT,ITYPE
      COMMON /SET11/ NCUST(2),MAXLN(2),DELAY(2),DELAYSQ(2),XMDELAY(2),
     -DELAYO(2),DELAYX(2),CLTIME(2),CLTIMSQ(2),CUSTIM(2),CTIMSQ(2),
     -PROBO(2),PROBL(2),LTOT(2),ISTOT(2)
      EQUIVALENCE (IBUFF(1),CLOCK(1)),(IBUFF(201),KIM(1)),
     -(IBUFF(301),LOG(1))
C
C     HISTORY ENTRY
C
      CLOCK(I3)=TIME-PSTIME
      KIM(I3)=K1+4
      I3=I3+1; NTOT=NTOT+1
      IF(I3.NE.101) GO TO 4
      CALL BUFFEROUT(7,1,IBUFF,400,IND)
    1 IF(IND.EQ.1) GO TO 1
      GO TO (1,3,2,2) IND
    2 OUTPUT(101) 'BUFFERING ERROR 4'
    3 I3=1
C
C     SERVER BUSY
C
    4 IBUSY(K1)=1
      II=I+2
C
C     SELECT QUEUE DISCIPLINE
C
      GO TO (10,20,30,40) IDSPLN(I)
C
C     FIFO
C
```

```
   10 STIME=TIM(1,II)
      WAIT=TIME-TIM(1,I)
      K=1
      GO TO 50
C
C     LIFO
C
   20 STIME=TIM(KP(I),II)
      WAIT=TIME-TIM(KP(I),I)
      K=KP(I)
      GO TO 50
C
C     SSTF
C
   30 STIME=TIM(1,II)
      K=1
      DO 35 J=2,KP(I)
   35 IF(TIM(J,II).LT.STIME) STIME=TIM(J,II); K=J
      WAIT=TIME-TIM(K,I)
      GO TO 50
C
C     RANDOM
C
   40 PIECE=1.0/KP(I)
      IX4=IX4*4099; RN=0.5+IX4*.5960464E-7
      K=1
      REPEAT 45 WHILE (K*PIECE).LT.RN
   45 K=K+1
      STIME=TIM(K,II)
      WAIT=TIME-TIM(K,I)
C
C     ADJUST ARRAY
C
   50 DO 55 J=K,KP(I)
      TIM(J,I)=TIM(J+1,I)
   55 TIM(J,II)=TIM(J+1,II)
      KP(I)=KP(I)-1
C
C     MINIMUM SERVICE TIME 1.2 SECONDS (TIME FOR MOTION)
C
      STIME=AMAX(STIME,1.2)
C
C     STATISTICS
C
   60 DELAY(I)=DELAY(I)+WAIT
      DELAYSQ(I)=DELAYSQ(I)+WAIT**2
      IF(WAIT.GT.XMDELAY(I)) XMDELAY(I)=WAIT
      IF(WAIT.GT.0.0) DELAYO(I)=DELAYO(I)+1
```

```
      IF(WAIT.GT.XD) DELAYX(I)=DELAYX(I)+1
      CLTIME(I)=CLTIME(I)+STIME
      CLTIMSQ(I)=CLTIMSQ(I)+STIME**2
C
C     SET END-OF-SERVICE-TIME
C
      ETIME(K1)=TIME+STIME
      LINE(I)=LINE(I)-1
      RETURN
      END
```

```
C     **************************************************************
C
C     SUBROUTINE ENDSRV
C
C     PURPOSE:  RECORD END-OF-SERVICE IN HISTORY AND DETERMINE
C               IF BEGIN SERVICE FOLLOWS
C
C     **************************************************************
C

      SUBROUTINE ENDSRV
      DIMENSION IBUFF(400)
      DIMENSION NO(6)
      DATA NO/6*0/
      COMMON /SET6/ CLOCK(100),KIM(100),LOG(100)
      COMMON /SET7/ ETIME(9),I,I3,K1,TIM(115,4)
      COMMON /SET9/ IDIST(4),RATE(4),KK(4),CV(4),NFDBK,IFDBK,IDSPLN(2),
     -IDUAL,IBUSY(6),KP(2),LINE(2),TIME,PSTIME,PTIME(2),RUNTIME,IX1,
     -IX2,IX3,IX4,XD,LQ,MAXQ,P
      COMMON /SET10/ NTOT,ITYPE
      COMMON /SET11/ NCUST(2),MAXLN(2),DELAY(2),DELAYSQ(2),XMDELAY(2),
     -DELAYO(2),DELAYX(2),CLTIME(2),CLTIMSQ(2),CUSTIM(2),CTIMSQ(2),
     -PROBO(2),PROBL(2),LTOT(2),ISTOT(2)
      EQUIVALENCE (IBUFF(1),CLOCK(1)),(IBUFF(201),KIM(1)),
     -(IBUFF(301),LOG(1))
C
C     SERVER NOT BUSY
C
      IBUSY(K1)=0
      IF(ITYPE.NE.4) GO TO 10
C
C     DETERMINE IF FEEDBACK
C
      IX3=IX3*4099: RN=0.5+IX3*.5960464E-7
      IF(RN.GT.P) IFDBK=1: NFDBK=NFDBK+1
C
C     HISTORY ENTRY
C
   10 CLOCK(I3)=TIME-PSTIME
C
C     ALTERNATE IMAGES
C
      IF(NO(K1).EQ.0) NO(K1)=1: GO TO 100
      NO(K1)=0
  100 KIM(I3)=K1+10+6*NO(K1)
      IF(IFDBK.EQ.1) LOG(I3)=2: GO TO 11
      IF(ITYPE.EQ.3) LOG(I3)=2: GO TO 11
```

```
      LOG(I3)=1
   11 I3=I3+1: NTOT=NTOT+1
      IF(I3.NE.101) GO TO 15
      CALL BUFFEROUT(7,1,IBUFF,400,IND)
   12 IF(IND.EQ.1) GO TO 12
      GO TO (12,14,13,13) IND
   13 OUTPUT(101) 'BUFFERING ERROR 3'
   14 I3=1
C
C     TIME STATISTICS
C
   15 CALL ACCUM
C
C     IN CASE OF BEGIN-SERVICE
C
      LOG(I3)=6
      IF(IFDBK.EQ.0) GO TO 17
C
C     CAUSES DELAY
C
      IF(LINE(1).EQ.0)  LOG(I3)=7
      CALL TIMES: KP(I)=KP(I)+1: CALL BEGSRV: KP(I)=KP(I)-1:
     -CALL ARRIV: RETURN
C
C     QUEUE EMPTY?
C
   17 IF(LINE(I).EQ.0) ETIME(K1)=99999.: GO TO 20
      CALL BEGSRV
   20 IF(ITYPE.NE.3) RETURN
C
C     FEEDBACK SERVICE TIME
C
      I=2: CALL TIMES
      IF(KP(2).NE.1) GO TO 19
C
C     FEEDBACK TO EMPTY QUEUE?
C
      TIM(KP(1)+1,1)=TIM(KP(2),2)
      TIM(KP(1)+1,3)=TIM(KP(2),4)
      KP(2)=KP(2)-1
      ETIME(7)=TIM(KP(1)+1,1)
   19 RETURN
      END
```

```
C       ************************************************************
C
C       SUBROUTINE ACCUM
C
C       PURPOSE:  GATHER TIME STATISTICS AND ADVANCE STATE CHANGE
C                 VARIABLES
C
C       ************************************************************
C

        SUBROUTINE ACCUM
        COMMON /SET7/ ETIME(9),I,I3,K1,TIM(115,4)
        COMMON /SET9/ IDIST(4),RATE(4),KK(4),CV(4),NFDBK,IFDBK,IDSPLN(2),
       -IDUAL,IBUSY(6),KP(2),LINE(2),TIME,PSTIME,PTIME(2),RUNTIME,IX1,
       -IX2,IX3,IX4,XD,LQ,MAXQ,P
        COMMON /SET10/ NTOT,ITYPE
        COMMON /SET11/ NCUST(2),MAXLN(2),DELAY(2),DELAYSQ(2),XMDELAY(2),
       -DELAYO(2),DELAYX(2),CLTIME(2),CLTIMSQ(2),CUSTIM(2),CTIMSQ(2),
       -PROBO(2),PROBL(2),LTOT(2),ISTOT(2)
        Z=TIME-PTIME(I)
        CUSTIM(I)=CUSTIM(I)+LINE(I)*Z
        CTIMSQ(I)=CTIMSQ(I)+(LINE(I)*Z)**2
        IF(LINE(I).GT.0) PROBO(I)=PROBO(I)+Z
        IF(LINE(I).GT.LQ) PROBL(I)=PROBL(I)+Z
C
C       ADVANCE STATE CHANGE VARIABLES TO NOW
C
        PTIME(I)=TIME
        PSTIME=TIME
        RETURN
        END
```

```
C         ******************************************************************
C
C         SUBROUTINE TIMES
C
C         PURPOSE:  TO GENERATE AND STORE ARRIVAL (OR IDLE) TIMES
C                   AND SERVICE TIMES
C
C         ******************************************************************
C


          SUBROUTINE TIMES
          COMMON /SET7/ ETIME(9),I,I3,K1,TIM(115,4)
          COMMON /SET9/ IDIST(4),RATE(4),KK(4),CV(4),NFDBK,IFDBK,IDSPLN(2),
         -IDUAL,IBUSY(6),KP(2),LINE(2),TIME,PSTIME,PTIME(2),RUNTIME,IX1,
         -IX2,IX3,IX4,XD,LQ,MAXQ,P
          COMMON /SET10/ NTOT,ITYPE
          COMMON /SET11/ NCUST(2),MAXLN(2),DELAY(2),DELAYSQ(2),XMDELAY(2),
         -DELAYO(2),DELAYX(2),CLTIME(2),CLTIMSQ(2),CUSTIM(2),CTIMSQ(2),
         -PROBO(2),PROBL(2),LTOT(2),ISTOT(2)
C
C         FIRST TIME THROUGH GETS ARRIVAL (OR IDLE) TIME
C         IDLE TIMES STORED IN COLUMN 2
C
          I2=I
C
C         FEEDBACK? SLIP INTO QUEUE TIME ARRAY
C
          IF(IFDBK.EQ.1) TIM(KP(1)+2,I2)=TIM(KP(1)+1,I2);
         -TIM(KP(1)+2,I2+2)=TIM(KP(1)+1,I2+2);
         -TIM(KP(1)+1,I2)=TIME; I2=I+2; GO TO 1
          IF(ITYPE.NE.5) GO TO 1
C
C         DUAL ARRIVAL STREAMS?
C
          IF(IDUAL.EQ.1.AND.I.EQ.2) TIM(KP(2)+1,2)=TIM(KP(1)+1,1); I2=I+2
C
C         SELECT DISTRIBUTION
C
        1 GO TO (10,20,30) IDIST(I2)
C
C         K-ERLANG
C
       10 U=1.0/(KK(I2)*RATE(I2))
          SV=0.0
          DO 11 J=1,KK(I2)
C
C         GENERATE RANDOM NUMBER BETWEEN 0 AND 1
```

```
C
      IX1=IX1*4099: RN=0.5+IX1*.5960464E-7
   11 SV=SV-U*ALOG(RN)
      GO TO 88
C
C     HYPEREXPONENTIAL
C
   20 IX1=IX1*4099:  RN1=0.5+IX1*.5960464E-7
      IX2=IX2*4099: RN2=0.5+IX2*.5960464E-7
      U=1.0/RATE(I2)
      P=1.0/(CV(I2)** 2)
      UX=0.2928932
      IF(P.GE.RN1) UX=(U-((1.0-P)*UX))/P
      SV=-UX*ALOG(RN2)
      GO TO 88
C
C     DEGENERATE
C
   30 SV=1.0/RATE(I2)
   88 IF(I2.NE.I) GO TO 89
C
C     MINIMUM INTER-ARRIVAL TIME=0.5 ( MOTION TIME)
C
     -GO TO 87
      IF(ITYPE.EQ.2 .OR. ITYPE.EQ.5) SV=AMAX(SV,0.5);
C
C     MINIMUM INTER-ARRIVAL TIME=0.3 (MOTION TIME)
C
      SV=AMAX(SV,0.3)
   87 TIM(KP(I)+1,I2)=SV+TIME
      I2=I+2
      GO TO 1
   89 TIM(KP(I)+1,I2)=SV
      IF(ITYPE.NE.3) GO TO 95
      IF(KP(2).EQ.0) GO TO 91
C
C     NEXT END-IDLE TIME AT TAIL OF ARRAY
C
      TEMP1=TIM(KP(I)+1,2)
      TEMP2=TIM(KP(I)+1,4)
      J=KP(I)
      REPEAT 90 WHILE TEMP1.GT.TIM(J,2) .AND. J.GT.0
      TIM(J+1,2)=TIM(J,2): TIM(J+1,4)=TIM(J,4)
   90 J=J-1
      TIM(J+1,2)=TEMP1: TIM(J+1,4)=TEMP2
   91 KP(2)=KP(2)+1
   95 RETURN
      END
```

```
C     ***************************************************************
C
C     SUBROUTINE STATISTICS
C
C     PURPOSE:  DISPLAY THE STATISTICS PAGE
C
C     ***************************************************************
C

      SUBROUTINE STATISTICS
      DIMENSION IWORD(11)
      REAL LT,LC
      COMMON /STAT/ QT(2),BT(2),LT(2),QC(2),BC(2),LC(2),
     -DC(2),SC(2),WC(2),DM(2),DO(2),DX(2),QM(2),QO(2),
     -QL(2),DV(2),SV(2),QV(2),PL,NC(2)
      COMMON /SET5/ IDEV,IER
      COMMON /SET10/ NTOT,ITYPE
    1 FORMAT(16X,'STATISTICS',18X)
    2 FORMAT(7X,'Q1',6X,'Q2',13X,'Q1',6X,'Q2',4X)
    3 FORMAT('QT',3X,F6.2,2X,F6.2,4X,'QC',3X,F6.2,2X,F6.2,2X)
    4 FORMAT('BT',3X,F6.2,2X,F6.2,4X,'BC',3X,F6.2,2X,F6.2,2X)
    5 FORMAT('LT',3X,F6.2,2X,F6.2,4X,'LC',3X,F6.2,2X,F6.2,2X)
    6 FORMAT('DC',3X,F6.2,2X,F6.2,4X,'DV',3X,F6.2,2X,F6.2,2X)
    7 FORMAT('SC',3X,F6.2,2X,F6.2,4X,'SV',3X,F6.2,2X,F6.2,2X)
    8 FORMAT('WC',3X,F6.2,2X,F6.2,4X,'QV',3X,F6.2,2X,F6.2,2X)
    9 FORMAT('DM',3X,F6.2,2X,F6.2,4X,'QM',3X,F6.2,2X,F6.2,2X)
   10 FORMAT('DO',3X,F6.2,2X,F6.2,4X,'QO',3X,F6.2,2X,F6.2,2X)
   11 FORMAT('DX',3X,F6.2,2X,F6.2,4X,'QL',3X,F6.2,2X,F6.2,2X)
   12 FORMAT('NC',3X,I4,4X,I4,6X,'PS',3X,F6.2,10X)
   13 FORMAT(12X,'ENTER  1. CONTINUE',14X)
   14 FORMAT(19X,'2. PARAMETER LISTING',5X)
      ENCODE(44,1,IWORD)
      CALL TEXTO (IDEV,IWORD,11, 6,7,2,3,IER)
      ENCODE(44,2,IWORD)
      CALL TEXTO (IDEV,IWORD,11, 8,7,2,3,IER)
      ENCODE(44, 3,IWORD) QT(1),QT(2),QC(1),QC(2)
      CALL TEXTO (IDEV,IWORD,11,11,7,2,3,IER)
      ENCODE(44, 4,IWORD) BT(1),BT(2),BC(1),BC(2)
      CALL TEXTO (IDEV,IWORD,11,13,7,2,3,IER)
      ENCODE(44, 5,IWORD) LT(1),LT(2),LC(1),LC(2)
      CALL TEXTO (IDEV,IWORD,11,15,7,2,3,IER)
      ENCODE(44, 6,IWORD) DC(1),DC(2),DV(1),DV(2)
      CALL TEXTO (IDEV,IWORD,11,18,7,2,3,IER)
      ENCODE(44, 7,IWORD) SC(1),SC(2),SV(1),SV(2)
      CALL TEXTO (IDEV,IWORD,11,20,7,2,3,IER)
      ENCODE(44, 8,IWORD) WC(1),WC(2),QV(1),QV(2)
      CALL TEXTO (IDEV,IWORD,11,22,7,2,3,IER)
```

```
ENCODE(44, 9,IWORD) DM(1),DM(2),QM(1),QM(2)
CALL TEXTO (IDEV,IWORD,11,25,7,2,3,IER)
ENCODE(44,10,IWORD) DO(1),DO(2),QO(1),QO(2)
CALL TEXTO (IDEV,IWORD,11,27,7,2,3,IER)
ENCODE(44,11,IWORD) DX(1),DX(2),QL(1),QL(2)
CALL TEXTO (IDEV,IWORD,11,29,7,2,3,IER)
ENCODE(44,12,IWORD) NC(1),NC(2),PL
CALL TEXTO (IDEV,IWORD,11,32,7,2,3,IER)
ENCODE(44,13,IWORD)
CALL TEXTO (IDEV,IWORD,11,34,7,2,3,IER)
ENCODE(44,14,IWORD)
CALL TEXTO (IDEV,IWORD,11,36,7,2,3,IER)
RETURN
END
```

```
C      ***********************************************************
C
C      SUBROUTINE PARAMETERS
C
C      PURPOSE:  DISPLAY THE CURRENT PARAMETERS
C
C      ***********************************************************
C

       SUBROUTINE PARAMETERS
       DIMENSION IWORD(11)
       COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
       COMMON /SET5/ IDEV,IER
       COMMON /SET8/ CHNG(10),ENDD(5,22),IFLAG(22),ITOT,IPOS(2),
      -ILN(2),IMAN(7,22),NSIM,NQ(2),ITQ(2,2),NPOP,ITPOP(2),XX(5,22),
      -XXX(5,22),YY(5,22),YYY(5,22),IMD(5)
       COMMON /SET9/ IDIST(4),RATE(4),KK(4),CV(4),NFDBK,IFDBK,IDSPLN(2),
      -IDUAL,IBUSY(6),KP(2),LINE(2),TIME,PSTIME,PTIME(2),RUNTIME,IX1,
      -IX2,IX3,IX4,XD,LQ,MAXQ,P
       COMMON /SET10/ NTOT,ITYPE
     1 FORMAT(7X,'ITYPE',5X,I4,6X,'RUNTIME  ',F5.2,3X)
     2 FORMAT(7X,'IC',8X,I4,6X,'IC1',7X,I4,3X)
     3 FORMAT(10X,'1',9X,'2',9X,'3',9X,'4',3X)
     4 FORMAT('IDIST  ',I4,6X,I4,6X,I4,6X,I4,3X)
     5 FORMAT('KK     'I4,6X,I4,6X,I4,6X,I4,3X)
     6 FORMAT('RATE   ',F5.2,5X,F5.2,5X,F5.2,5X,F5.2,3X)
     7 FORMAT('CV     ',F5.2,5X,F5.2,5X,F5.2,5X,F5.2,3X)
     8 FORMAT('IDSPLN ',I4,6X,I4,23X)
     9 FORMAT(7X,'LQ',8X,I4,6X,'MAXQ',6X,I4,3X)
    10 FORMAT(7X,'XD',7X,F5.2,6X,'NPOP',5X,I4,3X)
    11 FORMAT(7X,'IDUAL',5X,I4,6X,'P',8X,F5.2,3X)
    12 FORMAT(12X,'ENTER  1. CONTINUE',14X)
    13 FORMAT(19X,'2. STATISTICS',12X)
       ENCODE (44, 1,IWORD) ITYPE,RUNTIME
       CALL TEXTO (IDEV,IWORD,11, 8,7,2,3,IER)
       ENCODE (44, 2,IWORD) IC,IC1
       CALL TEXTO (IDEV,IWORD,11,10,7,2,3,IER)
       ENCODE (44, 3,IWORD)
       CALL TEXTO (IDEV,IWORD,11,13,7,2,3,IER)
       ENCODE (44, 4,IWORD) IDIST(1),IDIST(2),IDIST(3),IDIST(4)
       CALL TEXTO (IDEV,IWORD,11,15,7,2,3,IER)
       ENCODE (44, 5,IWORD) KK(1),KK(2),KK(3),KK(4)
       CALL TEXTO (IDEV,IWORD,11,17,7,2,3,IER)
       ENCODE (44, 6,IWORD) RATE(1),RATE(2),RATE(3),RATE(4)
       CALL TEXTO (IDEV,IWORD,11,19,7,2,3,IER)
       ENCODE (44, 7,IWORD) CV(1),CV(2),CV(3),CV(4)
       CALL TEXTO (IDEV,IWORD,11,21,7,2,3,IER)
```

```
ENCODE (44, 8.IWORD) IDSPLN(1),IDSPLN(2)
CALL TEXTO (IDEV,IWORD,11,23,7,2,3,IER)
ENCODE (44, 9,IWORD) LQ,MAXQ
CALL TEXTO (IDEV,IWORD,11,26,7,2,3,IER)
ENCODE (44,10,IWORD) XD,NPOP
CALL TEXTO (IDEV,IWORD,11,28,7,2,3,IER)
ENCODE (44,11,IWORD) IDUAL,P
CALL TEXTO (IDEV,IWORD,11,30,7,2,3,IER)
ENCODE (44,12,IWORD)
CALL TEXTO (IDEV,IWORD,11,33,7,2,3,IER)
ENCODE (44,13,IWORD)
CALL TEXTO (IDEV,IWORD,11,35,7,2,3,IER)
RETURN
END
```

```
C     ****************************************************************
C
C     SUBROUTINE DRIVER
C
C     PURPOSE:  TO CAUSE THE DYNAMIC DISPLAY OF THE TIME HISTORY
C
C     OTHER SUBROUTINES:  SETFLAGS, FLAGS, REMAN
C
C     ****************************************************************
C

      SUBROUTINE DRIVER
      DIMENSION IBUFF(400)
      COMMON /SET6/ CLOCK(100),KIM(100),LOG(100)
      COMMON /SET8/ CHNG(10),ENDD(5,22),IFLAG(22),ITOT,IPOS(2),
     -ILN(2),IMAN(7,22),NSIM,NQ(2),ITQ(2,2),NPOP,ITPOP(2),XX(5,22),
     -XXX(5,22),YY(5,22),YYY(5,22),IMD(5)
      COMMON /SET10/ NTOT,ITYPE
      COMMON /SET12/ IDONE,I3,IM
      EQUIVALENCE (IBUFF(1),CLOCK(1)),(IBUFF(201),KIM(1)),
     -(IBUFF(301),LOG(1))
   10 IDONE=1
      CALL BUFFERIN (7,1,IBUFF,400,IND)
   11 IF(IND.EQ.1) GO TO 11
      GO TO (11,13,12,12) IND
   12 OUTPUT(101) 'BUFFERIN ERROR '
   13 I3=1
C
C     DISALLOWS TOO SMALL A VALUE AFTER BUFFERIN
C
      IF(CLOCK(I3).LE.0.1) CLOCK(I3)=.1
C
C     SET MAX (HYBRID LIMITATION)
C
      IF(CLOCK(I3).GT.99.) CLOCK(I3)=99.
C
C     SET TIMER FOR NEXT EVENT
C
      CALL DAC(1,CLOCK(I3)/100.)
      CALL COMPUTE
      CALL ENABLE
C     WAIT
C
C
   19 IF(IDONE.EQ.1) GO TO 19
      CALL DISABLE
      IF(IDONE.EQ.2) GO TO 10
```

```
30 REWIND 7: RETURN
   END
```

```
C       *****************************************************************
C
C       SUBROUTINE SETFLAGS
C
C       PURPOSE:  SET IMAGE FLAGS AS PER TIME HISTORY AT THE
C                 OCCURANCE OF A TIMING INTERUPT (53 INTERUPT)
C
C       *****************************************************************
C

        SUBROUTINE SETFLAGS
        COMMON /SET6/ CLOCK(100),KIM(100),LOG(100)
        COMMON /SET8/ CHNG(10),ENDD(5,22),IFLAG(22),ITOT,IPOS(2),
       -ILN(2),IMAN(7,22),NSIM,NQ(2),ITQ(2,2),NPOP,ITPOP(2),XX(5,22),
       -XXX(5,22),YY(5,22),YYY(5,22),IMD(5)
        COMMON /SET10/ NTOT,ITYPE
        COMMON /SET12/ IDONE,I3,IM
C
C       SET CORRECT IMAGE FLAG TO MOVEMENT LOGIC VALUE AS PER HISTORY
C
     10 IFLAG(KIM(I3))=LOG(I3)
        I3=I3+1
        ITOT=ITOT+1
        IF(ITOT.GT.NTOT) IDONE=3; RETURN
        IF(I3.GT.100) IDONE=2; RETURN
C
C       IF TIME TO NEXT EVENT IS SMALL, SET THE FLAG
C
        IF(CLOCK(I3).LE.0.1) GO TO 10
C
C       SET MAX (HYBRID LIMITATION)
C
        IF(CLOCK(I3).GT.99.) CLOCK(I3)=99.
C
C       SET THE TIMER FOR THE NEXT EVENT INTERUPT
C
        CALL DAC(1,CLOCK(I3)/100.)
        CALL COMPUTE
        RETURN
        END
```

```
C       ************************************************************
C
C       SUBROUTINE FLAGS
C
C       PURPOSE:  DISPLAY AND MOVE THE DYNAMIC IMAGES
C
C       ************************************************************
C

        SUBROUTINE FLAGS
        COMMON /SET5/ IDEV, IER
        COMMON /SET8/ CHNG(10),ENDD(5,22),IFLAG(22),ITOT,IPOS(2),
       -ILN(2),IMAN(7,22),NSIM,NQ(2),ITQ(2,2),NPOP,ITPOP(2),XX(5,22),
       -XXX(5,22),YY(5,22),YYY(5,22),IMD(5)
        COMMON /SET10/ NTOT,ITYPE
        COMMON /SET12/ IDONE,I3,IM
      1 FORMAT(I4)
C
C       CHECK ALL THE FLAGS
C
        IM=22
        REPEAT 99 WHILE IM.GT.0
        IF(IFLAG(IM).GT.0) GO TO(10,20,30,40,50,60,70,
       -71,72,73,74,75) IFLAG(IM)
     99 IM=IM-1
        RETURN
C
C       HORIZONTAL MOVEMENT RIGHT (ARRIVALS, STRAIGHT DEPARTURES)
C
     10 IF(XX(1,IM).NE.XXX(1,IM) .OR. IM.LE.10) GO TO 5
        IF(IM.GE.17) IM=IM-12; CALL REMAN; IM=IM+12; GO TO 5
        IM=IM-6; CALL REMAN; IM=IM+6
      5 CALL GRAPHO(IDEV,IMAN(1,IM),7,NSIM+IM,IER)
        TEST=XX(1,IM)+CHNG(1)
        IF(TEST.LE.ENDD(5,IM)) GO TO 18
     11 IF(IM.GT.4) GO TO 13
     12 IF(ITYPE.NE.3) GO TO 15
        NPOP=NPOP-1; ENCODE(4,1,ITPOP) NPOP
        CALL TEXTO (IDEV,ITPOP,2,ILN(2),IPOS(2),3,3,IER)
        NQ(1)=NQ(1)+1
        IF(NQ(1).LE.0) GO TO 13
        ENCODE(4,1,ITQ(1,1)) NQ(1)
        CALL TEXTO (IDEV,ITQ(1,1),2,ILN(1),IPOS(1),3,3,IER)
        GO TO 13
     15 IF(ITYPE.NE.5) IT=1; GO TO 6
        IF(IM.EQ.1 .OR. IM.EQ.3) IT=1; GO TO 6
        IT=2
```

```
    6 NQ(IT)=NQ(IT)+1:
     -IF(NQ(IT).GT.0) ENCODE(4,1,ITQ(1,IT)) NQ(IT):
     -CALL TEXTO (IDEV,ITQ(1,IT),2,ILN(IT),IPOS(IT),3,3,IER)
   13 CALL REMAN: GO TO 99
   18 DO 19 I=2,6
      J=I-1
      XX(J,IM)=XX(J,IM)+CHNG(1)
   19 IMAN(I,IM)=IPACK(XX(J,IM),YY(J,IM),IMD(J))
      GO TO 99
C
C     HORIZONTAL MOVEMENT RIGHT (WHEN A COURSE CHANGE WILL OCCUR)
C
   20 IF(XX(1,IM).NE.XXX(1,IM) .OR. IM.LE.10) GO TO 7
      IF(IM.GE.17) IM=IM-12: CALL REMAN: IM=IM+12: GO TO 7
      IM=IM-6: CALL REMAN: IM=IM+6
    7 CALL GRAPHO (IDEV,IMAN(1,IM),7,NSIM+IM,IER)
      TEST=XX(1,IM)+CHNG(3)
      IF(TEST.GT.ENDD(1,IM)) IFLAG(IM)=3: GO TO 99
      DO 28 I=2,6
      J=I-1
      XX(J,IM)=XX(J,IM)+CHNG(3)
   28 IMAN(I,IM)=IPACK(XX(J,IM),YY(J,IM),IMD(J))
      GO TO 99
C
C     VERTICAL MOVEMENT DOWNWARD
C
   30 CALL GRAPHO (IDEV,IMAN(1,IM),7,NSIM+IM,IER)
      TEST=YY(1,IM)-CHNG(3)
      IF(TEST.GE.ENDD(2,IM)) GO TO 31
      IF(IM.LE.4) CALL REMAN: GO TO 99
      IFLAG(IM)=4: GO TO 99
   31 DO 38 I=2,6
      J=I-1
      YY(J,IM)=YY(J,IM)-CHNG(3)
   38 IMAN(I,IM)=IPACK(XX(J,IM),YY(J,IM),IMD(J))
      GO TO 99
C
C     HORIZONTAL MOVEMENT TO THE LEFT
C
   40 CALL GRAPHO (IDEV,IMAN(1,IM),7,NSIM+IM,IER)
      TEST=XX(1,IM)-CHNG(3)*2.0
      IF(TEST.GE.ENDD(3,IM)) GO TO 47
   45 IFLAG(IM)=5: GO TO 99
   47 DO 48 I=2,6
      J=I-1
      XX(J,IM)=XX(J,IM)-CHNG(3)*2.0
   48 IMAN(I,IM)=IPACK(XX(J,IM),YY(J,IM),IMD(J))
      GO TO 99
```

```
C
C     VERTICAL MOVEMENT UPWARD
C
   50 CALL GRAPHO (IDEV,IMAN(1,IM),7,NSIM+IM,IER)
      TEST=YY(1,IM)+CHNG(4)
      IF(TEST.LE.ENDD(4,IM)) GO TO 57
      IF(ITYPE.NE.4) GO TO 55
      NQ(1)=NQ(1)+1
      IF(NQ(1).LE.0) GO TO 56
      ENCODE(4,1,ITQ(1,1)) NQ(1)
      CALL TEXTO (IDEV,ITQ(1,1),2,ILN(1),IPOS(1),3,3,IER)
      GO TO 56
   55 NPOP=NPOP+1; ENCODE(4,1,ITPOP) NPOP;
     -CALL TEXTO (IDEV,ITPOP,2,ILN(2),IPOS(2),3,3,IER)
   56 CALL REMAN; GO TO 99
   57 DO 58 I=2,6
      J=I-1
      YY(J,IM)=YY(J,IM)+CHNG(4)
   58 IMAN(I,IM)=IPACK(XX(J,IM),YY(J,IM),IMD(J))
      GO TO 99
C
C     MOVEMENT TO-SERVICE
C
   60 IF(XX(1,IM).NE.XXX(1,IM)) GO TO 61
      IF(ITYPE.NE.5 .OR. IM.LT.8) GO TO 65
      NQ(2)=NQ(2)-1
      IF(NQ(2).GE.0) ENCODE(4,1,ITQ(1,2)) NQ(2);
     -CALL TEXTO (IDEV,ITQ(1,2),2,ILN(2),IPOS(2),3,3,IER)
      GO TO 61
   65 NQ(1)=NQ(1)-1
      IF(NQ(1).LT.0) GO TO 61
      ENCODE(4,1,ITQ(1,1)) NQ(1)
      CALL TEXTO (IDEV,ITQ(1,1),2,ILN(1),IPOS(1),3,3,IER)
   61 CALL GRAPHO (IDEV,IMAN(1,IM),7,NSIM+IM,IER)
      TEST=XX(1,IM)+CHNG(2)
      IF(TEST.GT.ENDD(1,IM)) IFLAG(IM)=0; GO TO 99
      DO 68 I=2,6
      J=I-1
      XX(J,IM)=XX(J,IM)+CHNG(2)
      YY(J,IM)=YY(J,IM)+CHNG(IM)
   68 IMAN(I,IM)=IPACK(XX(J,IM),YY(J,IM),IMD(J))
      GO TO 99
C
C     CAUSE DELAY
C
   70 IFLAG(IM)=8 ; GO TO 99
   71 IFLAG(IM)=9 ; GO TO 99
   72 IFLAG(IM)=10; GO TO 99
```

```
73  IFLAG(IM)=11: GO TO 99
74  IFLAG(IM)=12: GO TO 99
75  IFLAG(IM)=6 : GO TO 99
    END
```

```
C     ***********************************************************************
C
C     SUBROUTINE REMAN
C
C     PURPOSE:  BLANK OUT IMAGES, AND REPOSITION
C
C     ***********************************************************************
C

      SUBROUTINE REMAN
      COMMON /SET5/ IDEV, IER
      COMMON /SET8/ CHNG(10),ENDD(5,22),IFLAG(22),ITOT,IPOS(2),
     -ILN(2),IMAN(7,22),NSIM,NQ(2),ITQ(2,2),NPOP,ITPOP(2),XX(5,22),
     -XXX(5,22),YY(5,22),YYY(5,22),IMD(5)
      COMMON /SET10/ NTOT,ITYPE
      COMMON /SET12/ IDONE,I3,IM
      IF(IM.LE.4 .OR. IM.GE.11) IFLAG(IM)=0
      DO 10 K=2,6
   10 IMAN(K,IM)=0
      CALL GRAPHO (IDEV,IMAN(1,IM),7,NSIM+IM,IER)
      DO 20 K=2,6
      J=K-1
      XX(J,IM)=XXX(J,IM)
      YY(J,IM)=YYY(J,IM)
   20 IMAN(K,IM)=IPACK(XX(J,IM),YY(J,IM),IMD(J))
      RETURN
      END
```

```
C      ******************************************************************
C
C      SUBROUTINE GGC
C
C      PURPOSE:  DISPLAY STATIC PICTURE FOR G/G/C MODEL
C
C      ******************************************************************
C

       SUBROUTINE GGC
       COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
      -IPATH7(8)
       COMMON /SET2/ISERV2(19),IQUE1(7),IQUE2(12),IQUE3(7),IPATH4(10),
      -IPATH5(10),IPATH9(7),IPATH10(5),IPATH11(4),
      -IPATH12(4),IPATH13(6)
       COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
       COMMON /SET5/ IDEV,IER
       DO 10 I=1,IC
    10 MD(I)=1
       CALL SERV1
       CALL GRAPHO (IDEV,ISERV1,6*IC+1,1,IER)
       CALL PATH1
       CALL GRAPHO(IDEV,IPATH1,3*IC+1,2,IER)
       CALL PATH2
       CALL GRAPHO (IDEV,IPATH2,3*IC+1,3,IER)
       CALL GRAPHO (IDEV,IPATH12,4,4,IER)
       CALL GRAPHO (IDEV,IQUE1,7,5,IER)
       RETURN
       END
```

```
C       ************************************************************
C
C       SUBROUTINE LOSSES
C
C       PURPOSE:  DISPLAY STATIC PICTURE FOR G/G/C WITH LOSSES MODEL
C
C       ************************************************************
C

        SUBROUTINE LOSSES
        COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
       -IPATH7(8)
        COMMON /SET2/ISERV2(19),IQUE1(7),IQUE2(12),IQUE3(7),IPATH4(10),
       -IPATH5(10),IPATH9(7),IPATH10(5),IPATH11(4),
       -IPATH12(4),IPATH13(6)
        COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
        COMMON /SET5/ IDEV,IER
        CALL GGC
        CALL GRAPHO (IDEV,IPATH11,4,6,IER)
        RETURN
        END
```

```
C     ****************************************************************
C
C     SUBROUTINE FEEDBACK
C
C     PURPOSE:  DISPLAY STATIC PICTURE FOR G/G/C WITH FEEDBACK
C
C     ****************************************************************

      SUBROUTINE FEEDBK
C
      COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
     -IPATH7(8)
      COMMON /SET2/ISERV2(19),IQUE1(7),IQUE2(12),IQUE3(7),IPATH4(10),
     -IPATH5(10),IPATH9(7),IPATH10(5),IPATH11(4),
      IPATH12(4),IPATH13(6)
      COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
      COMMON /SET5/ IDEV,IER
      CALL GGC
      DO 10 I=1,4
   10 MD(I)=1
      IF(IC.LT.6)MD(1)=0;IF(IC.LT.4)MD(2)=0;IF(IC.LT.2)MD(3)=0
      CALL PATH7
      CALL GRAPHO (IDEV,IPATH7,8,6,IER)
      CALL GRAPHO (IDEV,IPATH10,5,7,IER)
      RFTURN
      END
```

```
C     ************************************************************
C
C     SUBROUTINE FINITE
C
C     PURPOSE:  DISPLAY STATIC PICTURE FOR G/G/C WITH FINITE SOURCE
C
C     ************************************************************
C

      SUBROUTINE FINITE
      COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
     -IPATH7(8)
      COMMON /SET2/ISERV2(19),IQUE1(7),IQUE2(12),IQUE3(7),IPATH4(10),
     -IPATH5(10),IPATH9(7),IPATH10(5),IPATH11(4),
     -IPATH12(4),IPATH13(6)
      COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
      COMMON /SET5/ IDEV,IER
      DO 10 I=1,4
   10 MD(I)=1
      IF(IC.LT.6)MD(1)=0:IF(IC.LT.4)MD(2)=0:IF(IC.LT.2)MD(3)=0
      CALL PATH7
      DO 11 I=1,IC
   11 MD(I)=1
      CALL SERV1
      CALL PATH1
      CALL PATH3
      CALL GRAPHO (IDEV,ISERV1,6*IC+1,1,IER)
      CALL GRAPHO (IDEV,IQUE1,7,2,IER)
      CALL GRAPHO (IDEV,IQUE3,7,3,IER)
      CALL GRAPHO (IDEV,IPATH1,3*IC+1,4,IER)
      CALL GRAPHO (IDEV,IPATH3,3*IC+1,5,IER)
      CALL GRAPHO (IDEV,IPATH7,8,6,IER)
      CALL GRAPHO (IDEV,IPATH9,7,7,IER)
      RETURN
      END
```

```
C      ****************************************************************
C
C      SUBROUTINE TWOGGC
C
C      PURPOSE:  DISPLAY STATIC PICTURE FOR 2 G/G/C MODEL
C
C      ****************************************************************
C


       SUBROUTINE TWOGGC
       COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
     -IPATH7(8)
       COMMON /SET2/ISERV2(19),IQUE1(7),IQUE2(12),IQUE3(7),IPATH4(10),
     -IPATH5(10),IPATH9(7),IPATH10(5),IPATH11(4),
     -IPATH12(4),IPATH13(6)
       COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
       COMMON /SET5/ IDEV,IER
       DO 10 I=1,6
    10 MD(I)=1
       IF(IC.LT.3) MD(6)=0; IF(IC.LT.2) MD(2)=0
       IF(IC1.LT.3) MD(1)=0; IF(IC1.LT.2) MD(5)=0
       CALL SERV1
       CALL PATH2
       CALL GRAPHO (IDEV,IQUE2,12,1,IER)
       CALL GRAPHO (IDEV,ISERV1,37,2,IER)
       CALL GRAPHO (IDEV,IPATH13,6,3,IER)
       CALL GRAPHO (IDEV,IPATH4,3*IC+1,4,IER)
       CALL GRAPHO (IDEV,IPATH5,3*IC1+1,5,IER)
       CALL GRAPHO (IDEV,IPATH2,19,6,IER)
       RETURN
       END
```

```
C     ************************************************************
C
C     SUBROUTINE SERV1
C
C     PURPOSE:  PACK THE SERVICE BOXES
C
C     ************************************************************
C

      SUBROUTINE SERV1
      COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
     -IPATH7(8)
      COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
      ISERV1(1)=IHEAD(0,10)
      ISERV1(2)=IPACK(X(11),Y(13),0)
      ISERV1(3)=IPACK(X(11),Y(17),MD(1))
      ISERV1(4)=IPACK(X(12),Y(17),MD(1))
      ISERV1(5)=IPACK(X(12),Y(13),MD(1))
      ISERV1(6)=IPACK(X(11),Y(13),MD(1))
      ISERV1(7)=0
      ISERV1(8)=IPACK(X(11),Y(11),0)
      ISERV1(9)=IPACK(X(11),Y(7),MD(2))
      ISERV1(10)=IPACK(X(12),Y(7),MD(2))
      ISERV1(11)=IPACK(X(12),Y(11),MD(2))
      ISERV1(12)=IPACK(X(11),Y(11),MD(2))
      ISERV1(13)=0
      ISERV1(14)=IPACK(X(11),Y(18),0)
      ISERV1(15)=IPACK(X(11),Y(20),MD(3))
      ISERV1(16)=IPACK(X(12),Y(20),MD(3))
      ISERV1(17)=IPACK(X(12),Y(18),MD(3))
      ISERV1(18)=IPACK(X(11),Y(18),MD(3))
      ISERV1(19)=0
      ISERV1(20)=IPACK(X(11),Y(6),0)
      ISERV1(21)=IPACK(X(11),Y(4),MD(4))
      ISERV1(22)=IPACK(X(12),Y(4),MD(4))
      ISERV1(23)=IPACK(X(12),Y(6),MD(4))
      ISERV1(24)=IPACK(X(11),Y(6),MD(4))
      ISERV1(25)=0
      ISERV1(26)=IPACK(X(11),Y(21),0)
      ISERV1(27)=IPACK(X(11),Y(23),MD(5))
      ISERV1(28)=IPACK(X(12),Y(23),MD(5))
      ISERV1(29)=IPACK(X(12),Y(21),MD(5))
      ISERV1(30)=IPACK(X(11),Y(21),MD(5))
      ISERV1(31)=0
      ISERV1(32)=IPACK(X(11),Y(3),0)
      ISERV1(33)=IPACK(X(11),Y(1),MD(6))
      ISERV1(34)=IPACK(X(12),Y(1),MD(6))
```

```
ISERV1(35)=IPACK(X(12),Y(3),MD(6))
ISERV1(36)=IPACK(X(11),Y(3),MD(6))
ISERV1(37)=0
RETURN
END
```

```
C      *****************************************************************
C
C      SUBROUTINE PATH1
C
C      PURPOSE:  PACK THE TO-SERVICE PATHS
C
C      *****************************************************************
C

       SUBROUTINE PATH1
       COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
      -IPATH7(8)
       COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
       IPATH1(1)=IHEAD(1,8)
       IPATH1(2)=IPACK(X(10),Y(12),0)
       IPATH1(3)=IPACK(X(11),Y(15),MD(1))
       IPATH1(4)=0
       IPATH1(5)=IPACK(X(10),Y(12),0)
       IPATH1(6)=IPACK(X(11),Y(9),MD(2))
       IPATH1(7)=0
       IPATH1(8)=IPACK(X(10),Y(12),0)
       IPATH1(9)=IPACK(X(11),Y(19),MD(3))
       IPATH1(10)=0
       IPATH1(11)=IPACK(X(10),Y(12),0)
       IPATH1(12)=IPACK(X(11),Y(5),MD(4))
       IPATH1(13)=0
       IPATH1(14)=IPACK(X(10),Y(12),0)
       IPATH1(15)=IPACK(X(11),Y(22),MD(5))
       IPATH1(16)=0
       IPATH1(17)=IPACK(X(10),Y(12),0)
       IPATH1(18)=IPACK(X(11),Y(2),MD(6))
       IPATH1(19)=0
       RETURN
       END
```

```
C      ************************************************************
C
C      SUBROUTINE PATH2
C
C      PURPOSE:  PACK THE HCRIZONTAL FROM-SERVICE PATHS
C
C      ************************************************************
C

       SUBROUTINE PATH2
       COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
      -IPATH7(8)
       COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
       IPATH2(1)=IHEAD(1,8)
       IPATH2(2)=IPACK(X(12),Y(15),0)
       IPATH2(3)=IPACK(X(14),Y(15),MD(1))
       IPATH2(4)=0
       IPATH2(5)=IPACK(X(14),Y(9),0)
       IPATH2(6)=IPACK(X(12),Y(9),MD(2))
       IPATH2(7)=0
       IPATH2(8)=IPACK(X(12),Y(19),0)
       IPATH2(9)=IPACK(X(14),Y(19),MD(3))
       IPATH2(10)=0
       IPATH2(11)=IPACK(X(14),Y(5),0)
       IPATH2(12)=IPACK(X(12),Y(5),MD(4))
       IPATH2(13)=0
       IPATH2(14)=IPACK(X(12),Y(22),0)
       IPATH2(15)=IPACK(X(14),Y(22),MD(5))
       IPATH2(16)=0
       IPATH2(17)=IPACK(X(14),Y(2),0)
       IPATH2(18)=IPACK(X(12),Y(2),MD(6))
       IPATH2(19)=0
       RETURN
       END
```

```
C     ************************************************************
C
C     SUBROUTINE PATH3
C
C     PURPOSE:  PACK THE SHORT HORIZONTAL FROM-SERVICE PATHS
C
C     ************************************************************
C

      SUBROUTINE PATH3
      COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
     -IPATH7(8)
      COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
      IPATH3(1)=IHEAD(1,8)
      IPATH3(2)=IPACK(X(12),Y(15),0)
      IPATH3(3)=IPACK(X(13),Y(15),MD(1))
      IPATH3(4)=0
      IPATH3(5)=IPACK(X(13),Y(9),0)
      IPATH3(6)=IPACK(X(12),Y(9),MD(2))
      IPATH3(7)=0
      IPATH3(8)=IPACK(X(12),Y(19),0)
      IPATH3(9)=IPACK(X(13),Y(19),MD(3))
      IPATH3(10)=0
      IPATH3(11)=IPACK(X(13),Y(5),0)
      IPATH3(12)=IPACK(X(12),Y(5),MD(4))
      IPATH3(13)=0
      IPATH3(14)=IPACK(X(12),Y(22),0)
      IPATH3(15)=IPACK(X(13),Y(22),MD(5))
      IPATH3(16)=0
      IPATH3(17)=IPACK(X(13),Y(2),0)
      IPATH3(18)=IPACK(X(12),Y(2),MD(6))
      IPATH3(19)=0
      RETURN
      END
```

```
C       ************************************************************
C
C       SUBROUTINE PATH7
C
C       PURPOSE:  PACK A VERTICAL PATH
C
C       ************************************************************
C

        SUBROUTINE PATH7
        COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
       -IPATH7(8)
        COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
        IPATH7(1)=IHEAD(1,8)
        IPATH7(2)=IPACK(X(13),Y(2),0)
        IPATH7(3)=IPACK(X(13),Y(5),MD(1))
        IPATH7(4)=IPACK(X(13),Y(9),MD(2))
        IPATH7(5)=IPACK(X(13),Y(15),MD(3))
        IPATH7(6)=IPACK(X(13),Y(19),1)
        IPATH7(7)=IPACK(X(13),Y(24),MD(4))
        IPATH7(8)=0
        RETURN
        END
```

```
C     ************************************************************
C
C     SUBROUTINE PACK
C
C     PURPOSE:  PACK OTHER BOXES AND PATHS
C
C     ************************************************************
C

      SUBROUTINE PACK
      COMMON /SET1/ISERV1(37),IPATH1(19),IPATH2(19),IPATH3(19),
     -IPATH7(8)
      COMMON /SET2/ISERV2(19),IQUE1(7),IQUE2(12),IQUE3(7),IPATH4(10),
     -IPATH5(10),IPATH9(7),IPATH10(5),IPATH11(4),
     -IPATH12(4),IPATH13(6)
      COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
      IQUE1(1)=IHEAD(0,10)
      IQUE1(2)=IPACK(X(9),Y(10),0)
      IQUE1(3)=IPACK(X(9),Y(14),1)
      IQUE1(4)=IPACK(X(10),Y(14),1)
      IQUE1(5)=IPACK(X(10),Y(10),1)
      IQUE1(6)=IPACK(X(9),Y(10),1)
      IQUE1(7)=0
      IQUE2(1)=IHEAD(0,10)
      IQUE2(2)=IPACK(X(9),Y(4),0)
      IQUE2(3)=IPACK(X(9),Y(6),1)
      IQUE2(4)=IPACK(X(10),Y(6),1)
      IQUE2(5)=IPACK(X(10),Y(4),1)
      IQUE2(6)=IPACK(X(9),Y(4),1)
      IQUE2(7)=IPACK(X(9),Y(18),0)
      IQUE2(8)=IPACK(X(9),Y(20),1)
      IQUE2(9)=IPACK(X(10),Y(20),1)
      IQUE2(10)=IPACK(X(10),Y(18),1)
      IQUE2(11)=IPACK(X(9),Y(18),1)
      IQUE2(12)=0
      IQUE3(1)=IHEAD(0,10)
      IQUE3(2)=IPACK(X(7),Y(8),0)
      IQUE3(3)=IPACK(X(3),Y(8),1)
      IQUE3(4)=IPACK(X(3),Y(16),1)
      IQUE3(5)=IPACK(X(7),Y(16),1)
      IQUE3(6)=IPACK(X(7),Y(8),1)
      IQUE3(7)=0
      ISERV2(1)=IHEAD(0,10)
      ISERV2(2)=IPACK(X(6),Y(18),0)
      ISERV2(3)=IPACK(X(4),Y(18),1)
      ISERV2(4)=IPACK(X(4),Y(20),1)
      ISERV2(5)=IPACK(X(6),Y(20),1)
```

```
ISERV2(6)=IPACK(X(6),Y(18),1)
ISERV2(7)=0
ISERV2(8)=IPACK(X(6),Y(17),0)
ISERV2(9)=IPACK(X(6),Y(13),1)
ISERV2(10)=IPACK(X(4),Y(13),1)
ISERV2(11)=IPACK(X(4),Y(17),1)
ISERV2(12)=IPACK(X(6),Y(17),1)
ISERV2(13)=0
ISERV2(14)=IPACK(X(6),Y(21),0)
ISERV2(15)=IPACK(X(6),Y(23),1)
ISERV2(16)=IPACK(X(4),Y(23),1)
ISERV2(17)=IPACK(X(4),Y(21),1)
ISERV2(18)=IPACK(X(6),Y(21),1)
ISERV2(19)=0
IPATH4(1)=IHEAD(1,8)
IPATH4(2)=IPACK(X(10),Y(5),0)
IPATH4(3)=IPACK(X(11),Y(5),1)
IPATH4(4)=0
IPATH4(5)=IPACK(X(10),Y(5),0)
IPATH4(6)=IPACK(X(11),Y(9),1)
IPATH4(7)=0
IPATH4(8)=IPACK(X(10),Y(5),0)
IPATH4(9)=IPACK(X(11),Y(2),1)
IPATH4(10)=0
IPATH5(1)=IHEAD(1,8)
IPATH5(2)=IPACK(X(10),Y(19),0)
IPATH5(3)=IPACK(X(11),Y(19),1)
IPATH5(4)=0
IPATH5(5)=IPACK(X(10),Y(19),0)
IPATH5(6)=IPACK(X(11),Y(22),1)
IPATH5(7)=0
IPATH5(8)=IPACK(X(10),Y(19),0)
IPATH5(9)=IPACK(X(11),Y(15),1)
IPATH5(10)=0
IPATH9(1)=IHEAD(1,8)
IPATH9(2)=IPACK(X(13),Y(24),0)
IPATH9(3)=IPACK(X(5),Y(24),1)
IPATH9(4)=IPACK(X(5),Y(16),1)
IPATH9(5)=IPACK(X(7),Y(12),0)
IPATH9(6)=IPACK(X(9),Y(12),1)
IPATH9(7)=0
IPATH10(1)=IHEAD(1,8)
IPATH10(2)=IPACK(X(13),Y(24),0)
IPATH10(3)=IPACK(X(8),Y(24),1)
IPATH10(4)=IPACK(X(8),Y(12),1)
IPATH10(5)=0
IPATH11(1)=IHEAD(1,8)
IPATH11(2)=IPACK(X(8),Y(12),0)
```

```
IPATH11(3)=IPACK(X(8),Y(25),1)
IPATH11(4)=0
IPATH12(1)=IHEAD(1,8)
IPATH12(2)=IPACK(X(9),Y(12),0)
IPATH12(3)=IPACK(X(1),Y(12),1)
IPATH12(4)=0
IPATH13(1)=IHEAD(1,8)
IPATH13(2)=IPACK(X(9),Y(5),0)
IPATH13(3)=IPACK(X(1),Y(5),1)
IPATH13(4)=IPACK(X(1),Y(19),0)
IPATH13(5)=IPACK(X(9),Y(19),1)
IPATH13(6)=0
RETURN
END
```

```
C     ******************************************************************
C
C     SUBROUTINE XES
C
C     PURPOSE:  POSITICN ALL DYNAMIC IMAGES
C
C     ******************************************************************
C

      SUBROUTINE XES
      COMMON /SET3/X(14),Y(25),MD(6),IC,IC1,NULL
      COMMON /SET5/ IDEV,IER
      COMMON /SET8/ CHNG(10),ENDD(5,22),IFLAG(22),ITOT,IPOS(2),
     -ILN(2),IMAN(7,22),NSIM,NQ(2),ITQ(2,2),NPOP,ITPOP(2),XX(5,22),
     -XXX(5,22),YY(5,22),YYY(5,22),IMD(5)
      COMMON /SETX/ A1,A2,A3,A4,G1,G2,G3,G4,S1,S2,S3,S4,S5,S6,S7,S8
      DO 10 I=1,22
      IMAN(1,I)=IHEAD(0,10)
      DO 9 J=2,7
    9 IMAN(J,I)=0
   10 CALL GRAPHO (IDEV,IMAN(1,I),7,NSIM+I,IER)
      XXX(1,01)=XX(1,01)=X(A1)+.02
      YYY(1,01)=YY(1,01)=Y(A3)-.05
      XXX(2,01)=XX(2,01)=X(A1)-.015
      YYY(2,01)=YY(2,01)=Y(A3)+.02
      XXX(3,01)=XX(3,01)=X(A1)
      YYY(3,01)=YY(3,01)=Y(A3)+.05
      XXX(4,01)=XX(4,01)=X(A1)+.015
      YYY(4,01)=YY(4,01)=Y(A3)+.02
      XXX(5,01)=XX(5,01)=X(A1)-.02
      YYY(5,01)=YY(5,01)=Y(A3)-.05
      XXX(1,02)=XX(1,02)=X(A2)+.02
      YYY(1,02)=YY(1,02)=Y(A4)-.05
      XXX(2,02)=XX(2,02)=X(A2)-.015
      YYY(2,02)=YY(2,02)=Y(A4)+.02
      XXX(3,02)=XX(3,02)=X(A2)
      YYY(3,02)=YY(3,02)=Y(A4)+.05
      XXX(4,02)=XX(4,02)=X(A2)+.015
      YYY(4,02)=YY(4,02)=Y(A4)+.02
      XXX(5,02)=XX(5,02)=X(A2)-.02
      YYY(5,02)=YY(5,02)=Y(A4)-.05
      XXX(1,03)=XX(1,03)=X(A1)+.02
      YYY(1,03)=YY(1,03)=Y(A3)-.05
      XXX(2,03)=XX(2,03)=X(A1)-.015
      YYY(2,03)=YY(2,03)=Y(A3)+.02
      XXX(3,03)=XX(3,03)=X(A1)
```

```
YYY(3,03)=YY(3,03)=Y(A3)+.05
XXX(4,03)=XX(4,03)=X(A1)+.015
YYY(4,03)=YY(4,03)=Y(A3)+.02
XXX(5,03)=XX(5,03)=X(A1)-.02
YYY(5,03)=YY(5,03)=Y(A3)-.05
XXX(1,04)=XX(1,04)=X(A2)+.02
YYY(1,04)=YY(1,04)=Y(A4)-.05
XXX(2,04)=XX(2,04)=X(A2)-.015
YYY(2,04)=YY(2,04)=Y(A4)+.02
XXX(3,04)=XX(3,04)=X(A2)
YYY(3,04)=YY(3,04)=Y(A4)+.05
XXX(4,04)=XX(4,04)=X(A2)+.015
YYY(4,04)=YY(4,04)=Y(A4)+.02
XXX(5,04)=XX(5,04)=X(A2)-.02
YYY(5,04)=YY(5,04)=Y(A4)-.05
XXX(1,05)=XX(1,05)=X(G1)+.02
YYY(1,05)=YY(1,05)=Y(G3)-.05
XXX(2,05)=XX(2,05)=X(G1)-.015
YYY(2,05)=YY(2,05)=Y(G3)+.02
XXX(3,05)=XX(3,05)=X(G1)
YYY(3,05)=YY(3,05)=Y(G3)+.05
XXX(4,05)=XX(4,05)=X(G1)+.015
YYY(4,05)=YY(4,05)=Y(G3)+.02
XXX(5,05)=XX(5,05)=X(G1)-.02
YYY(5,05)=YY(5,05)=Y(G3)-.05
XXX(1,06)=XX(1,06)=X(G1)+.02
YYY(1,06)=YY(1,06)=Y(G3)-.05
XXX(2,06)=XX(2,06)=X(G1)-.015
YYY(2,06)=YY(2,06)=Y(G3)+.02
XXX(3,06)=XX(3,06)=X(G1)
YYY(3,06)=YY(3,06)=Y(G3)+.05
XXX(4,06)=XX(4,06)=X(G1)+.015
YYY(4,06)=YY(4,06)=Y(G3)+.02
XXX(5,06)=XX(5,06)=X(G1)-.02
YYY(5,06)=YY(5,06)=Y(G3)-.05
XXX(1,07)=XX(1,07)=X(G1)+.02
YYY(1,07)=YY(1,07)=Y(G3)-.05
XXX(2,07)=XX(2,07)=X(G1)-.015
YYY(2,07)=YY(2,C7)=Y(G3)+.02
XXX(3,07)=XX(3,07)=X(G1)
YYY(3,07)=YY(3,07)=Y(G3)+.05
XXX(4,07)=XX(4,07)=X(G1)+.015
YYY(4,07)=YY(4,07)=Y(G3)+.02
XXX(5,07)=XX(5,07)=X(G1)-.02
YYY(5,07)=YY(5,07)=Y(G3)-.05
XXX(1,08)=XX(1,08)=X(G2)+.02
YYY(1,08)=YY(1,08)=Y(G4)-.05
XXX(2,08)=XX(2,08)=X(G2)-.015
```

```
YYY(2,08)=YY(2,08)=Y(G4)+.02
XXX(3,08)=XX(3,08)=X(G2)
YYY(3,08)=YY(3,08)=Y(G4)+.05
XXX(4,08)=XX(4,08)=X(G2)+.015
YYY(4,08)=YY(4,08)=Y(G4)+.02
XXX(5,08)=XX(5,08)=X(G2)-.02
YYY(5,08)=YY(5,08)=Y(G4)-.05
XXX(1,09)=XX(1,09)=X(G2)+.02
YYY(1,09)=YY(1,09)=Y(G4)-.05
XXX(2,09)=XX(2,09)=X(G2)-.015
YYY(2,09)=YY(2,09)=Y(G4)+.02
XXX(3,09)=XX(3,09)=X(G2)
YYY(3,09)=YY(3,09)=Y(G4)+.05
XXX(4,09)=XX(4,09)=X(G2)+.015
YYY(4,09)=YY(4,09)=Y(G4)+.02
XXX(5,09)=XX(5,09)=X(G2)-.02
YYY(5,09)=YY(5,09)=Y(G4)-.05
XXX(1,10)=XX(1,10)=X(G2)+.02
YYY(1,10)=YY(1,10)=Y(G4)-.05
XXX(2,10)=XX(2,10)=X(G2)-.015
YYY(2,10)=YY(2,10)=Y(G4)+.02
XXX(3,10)=XX(3,10)=X(G2)
YYY(3,10)=YY(3,10)=Y(G4)+.05
XXX(4,10)=XX(4,10)=X(G2)+.015
YYY(4,10)=YY(4,10)=Y(G4)+.02
XXX(5,10)=XX(5,10)=X(G2)-.02
YYY(5,10)=YY(5,10)=Y(G4)-.05
XXX(1,11)=XX(1,11)=X(S1)+.02
YYY(1,11)=YY(1,11)=Y(S3)-.05
XXX(2,11)=XX(2,11)=X(S1)-.015
YYY(2,11)=YY(2,11)=Y(S3)+.02
XXX(3,11)=XX(3,11)=X(S1)
YYY(3,11)=YY(3,11)=Y(S3)+.05
XXX(4,11)=XX(4,11)=X(S1)+.015
YYY(4,11)=YY(4,11)=Y(S3)+.02
XXX(5,11)=XX(5,11)=X(S1)-.02
YYY(5,11)=YY(5,11)=Y(S3)-.05
XXX(1,12)=XX(1,12)=X(S1)+.02
YYY(1,12)=YY(1,12)=Y(S4)-.05
XXX(2,12)=XX(2,12)=X(S1)-.015
YYY(2,12)=YY(2,12)=Y(S4)+.02
XXX(3,12)=XX(3,12)=X(S1)
YYY(3,12)=YY(3,12)=Y(S4)+.05
XXX(4,12)=XX(4,12)=X(S1)+.015
YYY(4,12)=YY(4,12)=Y(S4)+.02
XXX(5,12)=XX(5,12)=X(S1)-.02
YYY(5,12)=YY(5,12)=Y(S4)-.05
XXX(1,13)=XX(1,13)=X(S1)+.02
```

```
YYY(1,13)=YY(1,13)=Y(S5)-.05
XXX(2,13)=XX(2,13)=X(S1)-.015
YYY(2,13)=YY(2,13)=Y(S5)+.02
XXX(3,13)=XX(3,13)=X(S1)
YYY(3,13)=YY(3,13)=Y(S5)+.05
XXX(4,13)=XX(4,13)=X(S1)+.015
YYY(4,13)=YY(4,13)=Y(S5)+.02
XXX(5,13)=XX(5,13)=X(S1)-.02
YYY(5,13)=YY(5,13)=Y(S5)-.05
XXX(1,14)=XX(1,14)=X(S2)+.02
YYY(1,14)=YY(1,14)=Y(S6)-.05
XXX(2,14)=XX(2,14)=X(S2)-.015
YYY(2,14)=YY(2,14)=Y(S6)+.02
XXX(3,14)=XX(3,14)=X(S2)
YYY(3,14)=YY(3,14)=Y(S6)+.05
XXX(4,14)=XX(4,14)=X(S2)+.015
YYY(4,14)=YY(4,14)=Y(S6)+.02
XXX(5,14)=XX(5,14)=X(S2)-.02
YYY(5,14)=YY(5,14)=Y(S6)-.05
XXX(1,15)=XX(1,15)=X(S2)+.02
YYY(1,15)=YY(1,15)=Y(S7)-.05
XXX(2,15)=XX(2,15)=X(S2)-.015
YYY(2,15)=YY(2,15)=Y(S7)+.02
XXX(3,15)=XX(3,15)=X(S2)
YYY(3,15)=YY(3,15)=Y(S7)+.05
XXX(4,15)=XX(4,15)=X(S2)+.015
YYY(4,15)=YY(4,15)=Y(S7)+.02
XXX(5,15)=XX(5,15)=X(S2)-.02
YYY(5,15)=YY(5,15)=Y(S7)-.05
XXX(1,16)=XX(1,16)=X(S2)+.02
YYY(1,16)=YY(1,16)=Y(S8)-.05
XXX(2,16)=XX(2,16)=X(S2)-.015
YYY(2,16)=YY(2,16)=Y(S8)+.02
XXX(3,16)=XX(3,16)=X(S2)
YYY(3,16)=YY(3,16)=Y(S8)+.05
XXX(4,16)=XX(4,16)=X(S2)+.015
YYY(4,16)=YY(4,16)=Y(S8)+.02
XXX(5,16)=XX(5,16)=X(S2)-.02
YYY(5,16)=YY(5,16)=Y(S8)-.05
XXX(1,17)=XX(1,17)=X(S1)+.02
YYY(1,17)=YY(1,17)=Y(S3)-.05
XXX(2,17)=XX(2,17)=X(S1)-.015
YYY(2,17)=YY(2,17)=Y(S3)+.02
XXX(3,17)=XX(3,17)=X(S1)
YYY(3,17)=YY(3,17)=Y(S3)+.05
XXX(4,17)=XX(4,17)=X(S1)+.015
YYY(4,17)=YY(4,17)=Y(S3)+.02
XXX(5,17)=XX(5,17)=X(S1)-.02
```

```
YYY(5,17)=YY(5,17)=Y(S3)-.05
XXX(1,18)=XX(1,18)=X(S1)+.02
YYY(1,18)=YY(1,18)=Y(S4)-.05
XXX(2,18)=XX(2,18)=X(S1)-.015
YYY(2,18)=YY(2,18)=Y(S4)+.02
XXX(3,18)=XX(3,18)=X(S1)
YYY(3,18)=YY(3,18)=Y(S4)+.05
XXX(4,18)=XX(4,18)=X(S1)+.015
YYY(4,18)=YY(4,18)=Y(S4)+.02
XXX(5,18)=XX(5,18)=X(S1)-.02
YYY(5,18)=YY(5,18)=Y(S4)-.05
XXX(1,19)=XX(1,19)=X(S1)+.02
YYY(1,19)=YY(1,19)=Y(S5)-.05
XXX(2,19)=XX(2,19)=X(S1)-.015
YYY(2,19)=YY(2,19)=Y(S5)+.02
XXX(3,19)=XX(3,19)=X(S1)
YYY(3,19)=YY(3,19)=Y(S5)+.05
XXX(4,19)=XX(4,19)=X(S1)+.015
YYY(4,19)=YY(4,19)=Y(S5)+.02
XXX(5,19)=XX(5,19)=X(S1)-.02
YYY(5,19)=YY(5,19)=Y(S5)-.05
XXX(1,20)=XX(1,20)=X(S2)+.02
YYY(1,20)=YY(1,20)=Y(S6)-.05
XXX(2,20)=XX(2,20)=X(S2)-.015
YYY(2,20)=YY(2,20)=Y(S6)+.02
XXX(3,20)=XX(3,20)=X(S2)
YYY(3,20)=YY(3,20)=Y(S6)+.05
XXX(4,20)=XX(4,20)=X(S2)+.015
YYY(4,20)=YY(4,20)=Y(S6)+.02
XXX(5,20)=XX(5,20)=X(S2)-.02
YYY(5,20)=YY(5,20)=Y(S6)-.05
XXX(1,21)=XX(1,21)=X(S2)+.02
YYY(1,21)=YY(1,21)=Y(S7)-.05
XXX(2,21)=XX(2,21)=X(S2)-.015
YYY(2,21)=YY(2,21)=Y(S7)+.02
XXX(3,21)=XX(3,21)=X(S2)
YYY(3,21)=YY(3,21)=Y(S7)+.05
XXX(4,21)=XX(4,21)=X(S2)+.015
YYY(4,21)=YY(4,21)=Y(S7)+.02
XXX(5,21)=XX(5,21)=X(S2)-.02
YYY(5,21)=YY(5,21)=Y(S7)-.05
XXX(1,22)=XX(1,22)=X(S2)+.02
YYY(1,22)=YY(1,22)=Y(S8)-.05
XXX(2,22)=XX(2,22)=X(S2)-.015
YYY(2,22)=YY(2,22)=Y(S8)+.02
XXX(3,22)=XX(3,22)=X(S2)
YYY(3,22)=YY(3,22)=Y(S8)+.05
XXX(4,22)=XX(4,22)=X(S2)+.015
```

```
      YYY(4,22)=YY(4,22)=Y(S8)+.02
      XXX(5,22)=XX(5,22)=X(S2)-.02
      YYY(5,22)=YY(5,22)=Y(S8)-.05
      DO 20 I=1,22
      DO 20 J=2,6
      K=J-1
   20 IMAN(J,I)=IPACK(XXX(K,I),YYY(K,I),IMD(K))
      RETURN
      END
```

```
C       ****************************************************************
C
C       SUBROUTINE GINP
C
C       WRITTEN BY A. WONG, EE DEPT., USNPGS
C       PURPOSE:  PROVIDE NAMELIST INPUT FROM GRAPHICS TERMINAL
C
C       ****************************************************************
C

        SUBROUTINE GINP(IDEV,ITDIR,IBLK,IBUF)
        DIMENSION IBUF(1),ITDIR(1)
        IB=IBLK+1
        NULL=-1
        IF(IBUF(1))1,50,100
1       IF(IBUF(1).NE.-1)GO TO 100
        ENCODE(16,10,IBUF)
10      FORMAT('NAMELIST INPUT')
        CALL TEXTO(IDEV,IBUF,4,18,28,3,3,IER)
        IF(IER.NE.0)OUTPUT(101)IER,'GINP1'
        RETURN
 50     CALL TEXTO(IDEV,NULL,1,18,28,3,3,IER)
        IF(IER.NE.0)OUTPUT(101)IER,'NULL1'
        CALL TEXTO(IDEV,NULL,1,24,43,3,3,IER)
        IF(IER.NE.0)OUTPUT(101)IER,'NULL2'
        RETURN
 100    CALL TEXTR(IDEV,NULL,1,24,43,3,3,IER)
        IF(IER.NE.0)OUTPUT(101)IER,'GINP2'
110     IF(MOD(ITDIR(IB),8).EQ.0)GO TO 110
        CALL TEXTI(IDEV,IBUF,24,0,IB,IER)
        IF(IER.NE.0)OUTPUT(101)IER,'GINP3'
        RETURN
        END
```

```
C       ******************************************************************
C
C       SUBROUTINE GINPUT
C
C       WRITTEN BY A. WONG, EE DEPT., USNPGS
C       PURPOSE:  PROVIDE NAMELIST INPUT FROM GRAPHICS TERMINAL
C
C       ******************************************************************
C
$
$GINPUT PZE     0
        BRM     9SETUPN
        PZE     3
IDEV    PZE     0
ITDIR   PZE     0
IBLK    PZE     0
        LDP     IDEV
        STD     TEXTO&3
        LDA     *IBLK
        STA     BLOCK
        LDA     INPADR
        ADD     BUF
        STA     IBUF
        LDA     R>ONES
        STA     *IBUF
        BRM     TEXTO
        PZE     0
        LDA     INPADR
        ADD     READ
        STA     PATCH
        LDA     BRM
        XMA     *PATCH
        STA     BRM
        LDA     #101
        STA     *IBUF
INPADR  BRM     9INPUT
        STZ     *IBUF
        BRM     TEXTO
        PZE     0
        LDA     BRM
        XMA     *PATCH
        STA     BRM
        BRR     GINPUT
*
*
*
TEXTO   PZE     0
        BRM     GINP
```

```
         PZE     4
         PZE     0
         PZE     0
         PZE     BLOCK
IBUF     PZE     0
         MPO     TEXTO
         BRR     TEXTO
*
*
*
BLOCK    PZE     0
PATCH    PZE     0
BRM      BRM     TEXTO
READ     DATA    0563
BUF      DATA    0773
         END
$
```

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Chief of Naval Personnel Pers 11b Department of the Navy Washington, D. C. 20370	1
4.	Naval Postgraduate School Department of Operations Research and Administrative Sciences Monterey, California 93940	1
5.	Professor R. W. Butterworth, Code 55 Bd Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
6.	Professor W. M. Raike, Code 55 Rj Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
7.	Professor P. A. W. Lewis, Code 55 Lw Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
8.	Professor D. P. Gaver, Code 55 Gv Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
9.	Professor A. F. Andrus, Code 55 As Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	1

10. Lieutenant Cyrus Michael Riddell 1
1615 92 North East
Bellevue, Washington 98004

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

A Queuing Model Simulation with Dynamic Graphical Display

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Master's Thesis; March 1973

5. AUTHOR(S) (First name, middle initial, last name)

Cyrus Michael Riddell

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
March 1973	90	

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

The purpose of this thesis is to provide the student of queuing systems with a vehicle through which a better understanding of the interrelationships between stochastic processes and queuing systems can be achieved.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

A-31408

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Queues						
Simulation						
Computer Graphics						

Thesis 143254
R425 Riddell
c.1 A queuing model simulation with dynamic graphical display.